

IBM<sup>®</sup> DB2 Universal Database<sup>™</sup>



# Replication Guide and Reference

*Version 8*



IBM<sup>®</sup> DB2 Universal Database<sup>™</sup>



# Replication Guide and Reference

*Version 8*

Before using this information and the product it supports, be sure to read the general information under *Notices*.

This document contains proprietary information of IBM. It is provided under a license agreement and is protected by copyright law. The information contained in this publication does not include any product warranties, and any statements provided in this manual should not be interpreted as such.

You can order IBM publications online or through your local IBM representative.

- To order publications online, go to the IBM Publications Center at [www.ibm.com/shop/publications/order](http://www.ibm.com/shop/publications/order)
- To find your local IBM representative, go to the IBM Directory of Worldwide Contacts at [www.ibm.com/planetwide](http://www.ibm.com/planetwide)

To order DB2 publications from DB2 Marketing and Sales in the United States or Canada, call 1-800-IBM-4YOU (426-4968).

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© **Copyright International Business Machines Corporation 1994, 2002. All rights reserved.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

---

# Contents

<b>About this book</b>	<b>xi</b>
Who should read this book	xi
How to use this book	xi
Conventions	xiii
Terminology	xiii
How to read syntax diagrams	xiv
Road map	xv
How to send your comments	xvi

<b>What's new for DB2 replication for Version 8?</b>	<b>xvii</b>
Usability improvements	xvii
Performance improvements	xviii
New user interface	xix
New function	xx
Serviceability improvements	xxiii
Changes to replication system commands	xxiv
Changes to control tables	xxv
Functions no longer supported	xxvii

---

## Part 1. Replication guide . . . . . 1

<b>Chapter 1. Planning for replication</b>	<b>3</b>
Memory planning	3
Memory used by the Capture program	3
Memory used by the Apply program	5
Memory used by the Replication Alert Monitor	5
Storage planning	5
Planning log impact	6
Planning the storage requirements of target tables and control tables	8
Planning storage requirements for temporary files	9
Planning for conflict detection	11
Planning for non-DB2 relational sources	12
Planning transaction throughput rates for Capture triggers	12
Planning the log impact for non-DB2 relational source servers	12
Planning locks for Oracle source servers	12
Planning coexistence of pre-existing triggers with Capture triggers	13
Planning for code page translation	13

Replicating data between databases with compatible code pages	13
Configuring National Language Support (NLS) for Replication	14

## **Chapter 2. Setting up for replication . . . 15**

Controlling access to replication servers	15
Connectivity requirements for replication	15
Authorizing user IDs for replication	17
Authorization requirements for administration	17
Authorization requirements for the Capture program	19
Authorization requirements for Capture triggers on non-DB2 relational databases	20
Authorization requirements for the Apply program	21
Authorization requirements for the Replication Alert Monitor	22
Storing user IDs and passwords for replication (UNIX, Windows)	23
Setting up the replication control tables	23
Creating control tables (UNIX, Windows)	23
Creating control tables (z/OS)	24
Creating control tables (OS/400)	24
Creating control tables for non-DB2 relational sources	24
Creating multiple sets of Capture control tables	25
Setting up the replication programs	26
Setting up the replication programs (UNIX, Windows)	26
Setting up the Capture and Apply programs (OS/400)	30
Setting up the replication programs (z/OS)	32
Setting up journals (OS/400)	32
Creating journals for source tables (OS/400)	32
Managing journals and journal receivers (OS/400)	34

## **Chapter 3. Registering tables and views as replication sources . . . . . 37**

Registering DB2 tables as sources	37
-----------------------------------	----

Registering non-DB2 relational tables as sources. . . . .	39	Defining middle tiers in a multi-tier configuration. . . . .	85
Registration options for source tables . . . . .	41	Defining read-write targets (update-anywhere). . . . .	87
Registering a subset of columns (vertical subsetting) . . . . .	42	Using an existing table as the target table . . . . .	89
Full-refresh copying and change-capture replication . . . . .	42	Common properties for all target table types . . . . .	90
After-image columns and before-image columns . . . . .	44	Source columns that you want applied to the target . . . . .	91
Before-image prefix . . . . .	47	Source rows that you want applied to the target . . . . .	91
Stop the Capture program on error . . . . .	48	How source columns map to target columns . . . . .	92
How the Capture program stores updates . . . . .	48	Target key. . . . .	93
Preventing the recapture of changes (update-anywhere replication) . . . . .	49	How the Apply program updates the target key columns with the target-key change option . . . . .	94
Setting conflict detection (update-anywhere replication) . . . . .	54		
Registering tables that use remote journaling (OS/400) . . . . .	56	<b>Chapter 5. Replicating special data types</b> . . . . .	<b>97</b>
Using relative record numbers (RRN) instead of primary keys (OS/400) . . . . .	57	General data restrictions for replication . . . . .	97
How views behave as replication sources . . . . .	58	Replicating large objects . . . . .	98
Views over a single table. . . . .	58	Replicating DATALINK values. . . . .	99
Views over a join of two or more tables. . . . .	58	Setting up and using the ASNDLCOPY exit routine . . . . .	101
Registering views of tables as sources . . . . .	60	Setting up and using DLFM_ASCNOPYD (UNIX, Windows). . . . .	103
Maintaining CCD tables as sources (IMS) . . . . .	61	Setting up and using ASNDLCOPYD (OS/400). . . . .	105
<b>Chapter 4. Subscribing to sources.</b> . . . .	<b>63</b>		
Planning how to group sources and targets . . . . .	63	<b>Chapter 6. Subsetting data in your replication environment</b> . . . . .	<b>107</b>
Planning the number of subscription-set members . . . . .	64	Subsetting data during registration . . . . .	107
Planning the number of subscription sets per Apply qualifier. . . . .	65	Subsetting source data using views . . . . .	108
Creating subscription sets . . . . .	66	Defining triggers on CD tables to prevent specific rows from being captured (UNIX, Windows, z/OS) . . . . .	108
Processing options for subscription sets . . . . .	68	Subsetting data during subscription. . . . .	109
Specifying whether the set is active . . . . .	68		
Specifying how many minutes worth of data the Apply program retrieves . . . . .	69	<b>Chapter 7. Manipulating data in your replication environment</b> . . . . .	<b>111</b>
Deciding how the Apply program loads target tables that have referential integrity . . . . .	71	Enhancing data using stored procedures or SQL statements . . . . .	112
Specifying how the Apply program replicates changes for members in the set . . . . .	72	Mapping source and target columns that have different names. . . . .	113
Defining SQL statements or stored procedures for the subscription set . . . . .	73	Creating computed columns . . . . .	113
Scheduling the replication of a subscription set . . . . .	73		
Mapping source tables and views to target tables and views within a subscription set . . . . .	75	<b>Chapter 8. Customizing and running replication SQL scripts</b> . . . . .	<b>115</b>
Selecting a target type. . . . .	78		
Defining read-only target tables . . . . .	80	<b>Chapter 9. Operating the Capture program</b> . . . . .	<b>117</b>

Default operational parameters for the Capture program . . . . .	117	opt4one (UNIX, Windows, z/OS) . . . . .	145
Changing operational parameters for the Capture program . . . . .	119	pwdfile (UNIX, Windows) . . . . .	146
Starting the Capture program (UNIX, Windows, z/OS) . . . . .	121	sleep (UNIX, Windows, z/OS) . . . . .	146
autoprune (UNIX, Windows, z/OS) . . . . .	123	spillfile (UNIX, Windows, z/OS) . . . . .	147
autostop (UNIX, Windows, z/OS) . . . . .	124	sqlerrorcontinue (UNIX, Windows) . . . . .	147
capture_path (UNIX, Windows, z/OS) . . . . .	124	term (UNIX, Windows, z/OS) . . . . .	148
capture_schema (UNIX, Windows, z/OS) . . . . .	125	trlreuse (UNIX, Windows, z/OS) . . . . .	148
capture_server (UNIX, Windows, z/OS) . . . . .	126	Starting the Apply program (OS/400) . . . . .	149
commit_interval (UNIX, Windows, z/OS) . . . . .	126	Stopping the Apply program . . . . .	151
lag_limit (UNIX, Windows, z/OS) . . . . .	126	Modifying the ASNDONE exit routine (UNIX, Windows, z/OS) . . . . .	151
logreuse (UNIX, Windows, z/OS) . . . . .	126	Modifying the ASNDONE exit routine (OS/400) . . . . .	153
logstdout (UNIX, Windows, z/OS) . . . . .	127	Refreshing target tables using the ASNLOAD exit routine . . . . .	154
memory_limit (UNIX, Windows, z/OS) . . . . .	127	Refreshing target tables with the ASNLOAD exit routine (UNIX, Windows) . . . . .	155
monitor_interval (UNIX, Windows, z/OS) . . . . .	128	Refreshing target tables with the ASNLOAD exit routine (z/OS) . . . . .	156
monitor_limit (UNIX, Windows, z/OS) . . . . .	128	Customizing ASNLOAD exit behavior (UNIX, Windows, z/OS) . . . . .	157
prune_interval (UNIX, Windows, z/OS) . . . . .	128	Refreshing target tables with the ASNLOAD exit routine (OS/400) . . . . .	159
retention_limit (UNIX, Windows, z/OS) . . . . .	129		
sleep_interval (UNIX, Windows, z/OS) . . . . .	130		
startmode (UNIX, Windows, z/OS) . . . . .	130		
term (UNIX, Windows, z/OS) . . . . .	131		
trace_limit (UNIX, Windows, z/OS) . . . . .	131		
Starting the Capture program (OS/400) . . . . .	132		
Altering the behavior of a running Capture program . . . . .	132		
Changing the operating parameters in the Capture parameters table . . . . .	134		
Stopping the Capture program . . . . .	135		
Suspending Capture (UNIX, Windows, z/OS) . . . . .	136		
Resuming Capture (UNIX, Windows, z/OS) . . . . .	136		
Reinitializing Capture . . . . .	137		
<b>Chapter 10. Operating the Apply program</b> . . . . .	<b>139</b>		
Starting the Apply program (UNIX, Windows, z/OS) . . . . .	139	<b>Chapter 11. Monitoring replication</b> . . . . .	<b>161</b>
apply_path (UNIX, Windows, z/OS) . . . . .	141	Checking for current status of replication programs (UNIX, Windows, z/OS) . . . . .	161
apply_qual (UNIX, Windows, z/OS) . . . . .	141	Checking the status of the Capture and Apply journal jobs (OS/400) . . . . .	163
control_server (UNIX, Windows, z/OS) . . . . .	142	Reviewing historical data for trends. . . . .	163
copyonce (UNIX, Windows, z/OS) . . . . .	142	Reviewing Capture program messages . . . . .	165
db2_subsystem (z/OS) . . . . .	143	Examining Capture program throughput . . . . .	165
delay (UNIX, Windows, z/OS) . . . . .	143	Displaying latency of data processed by the Capture program. . . . .	166
errwait (UNIX, Windows, z/OS) . . . . .	143	Reviewing Apply program messages . . . . .	167
inamsg (UNIX, Windows, z/OS) . . . . .	144	Examining Apply program throughput . . . . .	167
loadxit (UNIX, Windows, z/OS) . . . . .	144	Displaying the average length of time taken to replicate transactions. . . . .	167
logreuse (UNIX, Windows, z/OS) . . . . .	144	Setting up automated monitoring of your replication environment. . . . .	168
logstdout (UNIX, Windows, z/OS) . . . . .	145	Creating Monitor control tables . . . . .	169
notify (UNIX, Windows, z/OS) . . . . .	145	Defining contact information for the Replication Alert Monitor . . . . .	170
		Selecting alert conditions for the Replication Alert Monitor . . . . .	171
		Starting the Replication Alert Monitor . . . . .	173
		Scheduling when to start the Replication Alert Monitor . . . . .	178
		Reinitializing the Replication Alert Monitor . . . . .	179

Stopping the Replication Alert Monitor	179
Monitoring the progress of the Capture program (OS/400)	180

## Chapter 12. Making changes to your replication environment . . . . .

Registering new objects	183
Changing registration attributes for registered objects	184
Adding columns to source tables	185
Stop capturing changes for registered objects	188
Reactivating registrations	189
Removing registrations	190
Changing Capture schemas	191
Creating new subscription sets	194
Adding new subscription-set members to existing subscription sets (UNIX, Windows, z/OS)	195
Changing attributes of subscription sets	198
Changing subscription set names	199
Splitting a subscription set	201
Merging subscription sets	206
Changing Apply qualifiers of subscription sets	209
Deactivating subscription sets	212
Removing subscription sets	213
Coordinating replication events with database application events	214
Setting an event END_SYNCHPOINT using the USER type signal	214
Using the Capture CMD STOP signal	216
Performing a CAPSTART handshake signal outside of the Apply program	219
Performing a CAPSTOP signal	220
Promoting your replication configuration to another system	221

## Chapter 13. Maintaining your replication environment . . . . .

Maintaining your source systems	225
Maintaining source objects	225
Maintaining and retaining source logs and journal receivers	226
Maintaining your control tables	231
Using the RUNSTATS utility (UNIX, Windows, z/OS)	231
Rebinding packages and plans (UNIX, Windows, z/OS)	232
Reorganizing your control tables	232
Pruning your control tables	233

Preventing replication failures and recovering from errors	237
Maintaining your target tables	239

## Part 2. Replication Center . . . . .

### Chapter 14. Using the DB2 Replication Center . . . . .

Prerequisites for the DB2 Replication Center	245
Starting the DB2 Replication Center	245
Using the Replication Center launchpad	246
Managing user IDs and passwords for the Replication Center	247
Creating replication profiles	248
Creating control-table profiles	249
Creating source-object profiles	250
Creating target-object profiles	250
Creating replication control tables	251
Creating Capture control tables	252
Creating Apply control tables	253
Creating Monitor control tables	253
Adding servers to the Replication Center	254
Enabling a database for change capture (UNIX and Windows)	255
Registering sources	256
Creating subscription sets	257
Defining the information for the subscription set	258
Mapping sources to targets	259
Scheduling the subscription set	260
Adding SQL statements or stored procedures to the subscription set	260
Activating or deactivating subscription sets	261
Promoting replication objects	262
Promoting registered tables or views	262
Promoting subscription sets	263
Forcing a full refresh of target tables	263
Removing or deleting replication definitions	264
Operating the Capture program	265
Operating the Apply program	266
Operating the Replication Alert Monitor	266

### Chapter 15. Basic data replication scenario: DB2 for Windows . . . . .

Before you begin	269
Planning this scenario	270
Replication source	270
Replication target	271
Replication options	271



Setting up the replication environment for this scenario . . . . .	272
Step 1: Create replication control tables for the Capture program . . . . .	272
Step 2: Enable the source database for replication . . . . .	273
Step 3: Register a replication source . . . . .	274
Step 4: Create replication control tables for the Apply program . . . . .	277
Step 5: Create a subscription set and a subscription-set member . . . . .	277
Step 6: Create an Apply password file . . . . .	283
Step 7: Replicate the scenario data . . . . .	284
Operating in a replication environment . . . . .	286
Step 1: Update the source table . . . . .	286
Step 2: View status for the Capture program . . . . .	287
Step 3: View status for the Apply program . . . . .	288
Step 4: Stop the Capture and Apply programs . . . . .	289
Monitoring replication . . . . .	290
Step 1: Create replication control tables for the Monitor program . . . . .	291
Step 2: Create a contact for replication alerts . . . . .	293
Step 3: Select alert conditions for the Capture program . . . . .	293
Step 4: Select alert conditions for the Apply program . . . . .	294
Step 5: Start the Replication Alert Monitor for a monitor qualifier . . . . .	296

## Part 3. Replication reference. . . 299

### Chapter 16. Naming rules for replication objects . . . . . 301

### Chapter 17. System commands for replication (UNIX, Windows, z/OS) . . . 303

asnacmd: Operating Apply (UNIX, Windows, z/OS) . . . . .	304
asnanalyze: Operating the Analyzer (UNIX and Windows) . . . . .	305
asnapply: Starting Apply (UNIX, Windows, z/OS) . . . . .	308
asncap: Starting Capture (UNIX, Windows, z/OS) . . . . .	316

asnccmd: Operating Capture (UNIX, Windows, z/OS) . . . . .	322
asnmcmd: Operating the Replication Alert Monitor (UNIX, Windows, z/OS) . . . . .	329
asnmon: Starting the Replication Alert Monitor (UNIX, Windows, z/OS) . . . . .	330
asnpwd: Maintaining password files (UNIX and Windows) . . . . .	335
asnscrt: Creating a DB2 Replication Service to start Capture, Apply, or the Replication Alert Monitor (Windows only) . . . . .	338
asnsdrop: Dropping a DB2 Replication Service (Windows only) . . . . .	340
asntrc: Operating the replication trace facility (UNIX, Windows, z/OS) . . . . .	341

### Chapter 18. System commands for replication (OS/400) . . . . . 349

ADDDPRREG: Adding a DPR registration (OS/400) . . . . .	349
ADDDPRSUB: Adding a DPR subscription set (OS/400) . . . . .	359
ADDDPRSUBM: Adding a DPR subscription-set member (OS/400) . . . . .	376
ANZDPR: Operating the Analyzer (OS/400) . . . . .	387
CHGDPRCAPA: Changing DPR Capture attributes (OS/400) . . . . .	391
CRTDPRTBL: Creating the replication control tables (OS/400) . . . . .	396
ENDDPRAPY: Stopping Apply (OS/400) . . . . .	397
ENDDPRCAP: Stopping Capture (OS/400) . . . . .	400
GRTDPRAUT: Authorizing users (OS/400) . . . . .	402
INZDPRCAP: Reinitializing DPR Capture (OS/400) . . . . .	412
OVRDPRCAPA: Overriding DPR capture attributes (OS/400) . . . . .	414
RMVDPRREG: Removing a DPR registration (OS/400) . . . . .	420
RMVDPRSUB: Removing a DPR subscription set (OS/400) . . . . .	421
RMVDPRSUBM: Removing a DPR subscription-set member (OS/400) . . . . .	423
RVKDPRAUT: Revoking authority (OS/400) . . . . .	425
STRDPRAPY: Starting Apply (OS/400) . . . . .	427
STRDPRCAP: Starting Capture (OS/400) . . . . .	436
WRKDPTRTC: Using the DPR trace facility (OS/400) . . . . .	446

### Chapter 19. Operating the replication programs (z/OS) . . . . . 453

Using JCL or system-started tasks to operate the replication programs (z/OS) . . . . .	453
Using JCL to operate replication programs . . . . .	453
Using system-started tasks to operate replication programs . . . . .	454
Using MVS Automatic Restart Manager (ARM) to automatically restart replication programs (z/OS) . . . . .	455
Migrating your replication environment to data-sharing mode (z/OS) . . . . .	456

<b>Chapter 20. Using the Windows Service Control Manager to issue system commands (Windows) . . . . .</b>	<b>459</b>
Creating a replication service . . . . .	459
Operating a replication service . . . . .	461
Dropping a replication service . . . . .	461

<b>Chapter 21. Scheduling replication programs on various operating systems . 463</b>	
Scheduling programs on UNIX operating systems . . . . .	463
Scheduling programs on Windows operating systems . . . . .	463
Scheduling programs on z/OS operating systems . . . . .	464
Scheduling programs on the OS/400 operating system . . . . .	464

<b>Chapter 22. How the DB2 replication components communicate . . . . .</b>	<b>465</b>
The Replication Center, the Capture program or triggers, and the Apply program . . . . .	465
The Capture program and the Apply program . . . . .	466
The Capture triggers and the Apply program . . . . .	468
The Replication Center and the Replication Alert Monitor . . . . .	469
The Replication Alert Monitor, the Capture program, and the Apply program . . . . .	470

<b>Chapter 23. Table structures . . . . .</b>	<b>471</b>
Tables at a glance . . . . .	472
List of tables used at the Capture control server . . . . .	476
List of tables used at the Apply control server . . . . .	479
List of tables used at the Monitor control server . . . . .	481

List of tables used at the target server . . . . .	482
Tables at the Capture control server and their column descriptions . . . . .	483
ASN.IBMSNAP_CAPSCHEMAS . . . . .	483
schema.IBMSNAP_AUTHTKN (OS/400) . . . . .	484
schema.IBMSNAP_CAPENQ (UNIX, Windows, z/OS) . . . . .	485
schema.IBMSNAP_CAPMON . . . . .	486
schema.IBMSNAP_CAPPARMS . . . . .	487
schema.IBMSNAP_CAPTRACE (DB2 only) . . . . .	490
schema.CCD_table (non-DB2) . . . . .	491
schema.CD_table . . . . .	493
schema.IBMSNAP_PRUNCNTL . . . . .	494
schema.IBMSNAP_PRUNE_LOCK . . . . .	496
schema.IBMSNAP_PRUNE_SET . . . . .	496
schema.IBMSNAP_REG_EXT (OS/400) . . . . .	497
schema.IBMSNAP_REGISTER . . . . .	498
schema.IBMSNAP_REG_SYNCH (non-DB2 relational) . . . . .	505
schema.IBMSNAP_RESTART . . . . .	505
schema.IBMSNAP_SEQTABLE (Informix) . . . . .	507
schema.IBMSNAP_SIGNAL . . . . .	507
schema.IBMSNAP_UOW . . . . .	510

Tables at the Apply control server and their column descriptions . . . . .	513
ASN.IBMSNAP_APPENQ . . . . .	513
ASN.IBMSNAP_APPLY_JOB (OS/400) . . . . .	513
ASN.IBMSNAP_APPLYTRACE . . . . .	514
ASN.IBMSNAP_APPLYTRAIL . . . . .	515
ASN.IBMSNAP_SUBS_COLS . . . . .	519
ASN.IBMSNAP_SUBS_EVENT . . . . .	521
ASN.IBMSNAP_SUBS_MEMBR . . . . .	522
ASN.IBMSNAP_SUBS_SET . . . . .	526
ASN.IBMSNAP_SUBS_STMTS . . . . .	531

Tables at the Monitor control server and their column descriptions . . . . .	533
ASN.IBMSNAP_ALERTS . . . . .	533
ASN.IBMSNAP_CONDITIONS . . . . .	534
ASN.IBMSNAP_CONTACTGRP . . . . .	537
ASN.IBMSNAP_CONTACTS . . . . .	537
ASN.IBMSNAP_GROUPS . . . . .	538
ASN.IBMSNAP_MONENQ . . . . .	538
ASN.IBMSNAP_MONSERVERS . . . . .	539
ASN.IBMSNAP_MONTRACE . . . . .	540
ASN.IBMSNAP_MONTRAIL . . . . .	540

Tables at the target server and their column descriptions . . . . .	541
Base aggregate table . . . . .	541
Change aggregate table . . . . .	542
Consistent-change data (CCD) table . . . . .	543

Point-in-time table . . . . .	546	<b>Glossary . . . . .</b>	<b>649</b>
Replica table . . . . .	546	<b>Index . . . . .</b>	<b>661</b>
User copy table . . . . .	547	<b>Notices . . . . .</b>	<b>677</b>
<b>Chapter 24. Replication messages . . . .</b>	<b>549</b>	Trademarks . . . . .	680
<b>Appendix A. UNICODE and ASCII</b>		<b>Contacting IBM . . . . .</b>	<b>683</b>
<b>encoding schemes on z/OS. . . . .</b>	<b>641</b>	Product information . . . . .	683
Choosing an Encoding Scheme . . . . .	641		
Setting Encoding Schemes . . . . .	642		
<b>Appendix B. How the Capture program</b>			
<b>processes journal entry types (iSeries). .</b>	<b>643</b>		
<b>Appendix C. Starting the replication</b>			
<b>programs from within an application</b>			
<b>(UNIX, Windows) . . . . .</b>	<b>647</b>		



---

## About this book

This book describes how to plan, set up, and maintain a DB2® data replication environment.

The DB2 DataPropagator™ product is the focus of the book. You can use it with other products in the IBM® replication solution to tailor a replication environment that suits your business needs.

---

## Who should read this book

This book is written for database administrators, LAN administrators, and others who must set up and maintain a data replication environment. You should be familiar with standard database terminology, have a working knowledge of the operating systems that will be involved in replication, and have experience with database design, database administration, database security, server connectivity, and networking. You should understand the applications in your environment and how they manipulate the data that you want to replicate. You should be familiar with basic replication concepts and components.

---

## How to use this book

Most sections in this book pertain to replication function on all platforms. There are some sections that contain platform-specific information.

The organization and content of this book have changed since the last release. This book contains three parts:

- Part 1: *Replication guide* describes how to plan, set up, run, and maintain your replication environment. It includes the following chapters:
  - Chapter 1, “Planning for replication” on page 3 describes how to plan and design your replication environment.
  - Chapter 2, “Setting up for replication” on page 15 describes how to prepare your environment for replication.
  - Chapter 3, “Registering tables and views as replication sources” on page 37 describes what you need to know to register replication sources.
  - Chapter 4, “Subscribing to sources” on page 63 describes what you need to know to create subscription sets and add members to subscription sets.
  - Chapter 5, “Replicating special data types” on page 97 describes the replication options for LOB and DATALINK values in source tables.

- Chapter 6, “Subsetting data in your replication environment” on page 107 describes how to customize what data is captured and applied to the target as well as how the data is applied to the target.
- Chapter 7, “Manipulating data in your replication environment” on page 111 describes how to use the Capture program or the Apply program to manipulate source data.
- Chapter 8, “Customizing and running replication SQL scripts” on page 115 describes how to run SQL in your replication environment.
- Chapter 9, “Operating the Capture program” on page 117 describes how to operate the Capture program for all operating-system environments.
- Chapter 10, “Operating the Apply program” on page 139 describes how to operate the Apply program for all operating-system environments.
- Chapter 11, “Monitoring replication” on page 161 describes how to monitor your replication environment.
- Chapter 12, “Making changes to your replication environment” on page 183 describes how to make day-to-day changes in your replication environment.
- Chapter 13, “Maintaining your replication environment” on page 225 explains how to maintain your source systems, control tables, and target tables.
- Part 2: *Replication Center* describes the graphical user interface for replication. It includes the following chapters:
  - Chapter 14, “Using the DB2 Replication Center” on page 243 describes the Replication Center.
  - Chapter 15, “Basic data replication scenario: DB2 for Windows” on page 269 describes how to use the Replication Center to perform a simple replication scenario on sample data.
- Part 3: *Replication reference* describes replication commands, replication table structures, and replication messages. It includes the following chapters:
  - Chapter 16, “Naming rules for replication objects” on page 301 describes how to specify valid names for replication objects.
  - Chapter 17, “System commands for replication (UNIX, Windows, z/OS)” on page 303 describes commands that experienced DB2 replication users can use instead of the Replication Center for operating replication on UNIX, Windows, and z/OS operating systems.
  - Chapter 18, “System commands for replication (OS/400)” on page 349 describes the commands that you can use if you want to set up, administer, and maintain replication locally on the OS/400 operating system.
  - Chapter 19, “Operating the replication programs (z/OS)” on page 453 describes how to start and operate the replication programs using JCL or system-started tasks on z/OS.

- Chapter 20, “Using the Windows Service Control Manager to issue system commands (Windows)” on page 459 describes how to create services to operate the Replication programs on Windows NT operating systems.
- Chapter 21, “Scheduling replication programs on various operating systems” on page 463 describes how to schedule the Capture, Apply, and Replication Alert Monitor programs on various operating systems.
- Chapter 22, “How the DB2 replication components communicate” on page 465 describes how the replication components use the control tables to communicate with each other.
- Chapter 23, “Table structures” on page 471 describes the table structures for the replication tables that reside on the various replication servers.
- Chapter 24, “Replication messages” on page 549 describes the messages that are issued for replication for UNIX, Windows, and z/OS.
- Appendixes contain supplemental information that you might find useful.

---

## Conventions

This book uses these highlighting conventions:

- **Boldface type** indicates commands or user interface controls such as names of fields, folders, icons, or menu choices.
- Monospace type indicates examples of text that you enter exactly as shown.
- *Italic type* indicates variables that you should replace with a value. It is also used to indicate book titles and for emphasis of words.

---

## Terminology

This book uses standard terminology for database, connectivity, copying, SQL, and LAN concepts. All the replication concepts used in this book are defined in the glossary.

Unless otherwise specified, the following meanings are assumed:

**UNIX** UNIX refers to DB2 Universal Database for all UNIX platforms (such as UNIX, HP UX, and AIX), and DB2 Universal Database for Linux.

### Windows

Windows refers to DB2 Universal Database for Windows.

### OS/400

OS/400 refers to DB2 DataPropagator for iSeries.

**z/OS** z/OS refers to DB2 Universal Database for z/OS and OS/390. z/OS is the next generation of the OS/390 operating system, and it also includes UNIX System Services (USS) on z/OS.

## iSeries

iSeries refers to both iSeries and iSeries servers. iSeries is the next generation of AS/400 servers. The OS/400 operating system runs on both iSeries and iSeries servers.

For example, the section entitled *Starting the Apply program (UNIX, Windows, z/OS)* explains how to start the Apply program from DB2 Universal Database for Linux and for all UNIX platforms, DB2 Universal Database for Windows, or DB2 Universal Database for z/OS and OS/390. Also, the section entitled *Starting the Apply program (OS/400)* explains how to start the Apply program if you are using DB2 DataPropagator for iSeries.

---

## How to read syntax diagrams

The following rules apply to the syntax diagrams used in this book:

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

The ►— symbol indicates the beginning of a statement.

The —► symbol indicates that the statement syntax is continued on the next line.

The ►— symbol indicates that a statement is continued from the previous line.

The —►◄ symbol indicates the end of a statement.

Diagrams of syntactical units other than complete statements start with the ►— symbol and end with the —► symbol.

- Keywords, their allowable synonyms, and reserved parameters, are either shown in uppercase or lowercase, depending on the operating system. These items must be entered exactly as shown. Variables appear in lowercase italics (for example, *column-name*). They represent user-defined parameters or suboptions.

When entering commands, separate the parameters and keywords by at least one space if there is no intervening punctuation.

- Enter punctuation marks (slashes, commas, periods, parentheses, quotation marks, equal signs, and so on) and numbers exactly as given.
- Footnotes are shown by a number in parentheses, for example, (1).
- Required items appear on the horizontal line (the main path).

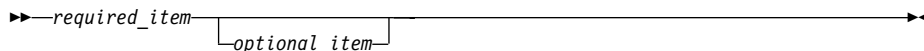
►—*required\_item*—►

- A parameter's default value is displayed above the path:

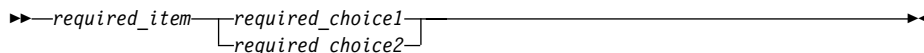
►—*required\_item*—<sup>*default\_value*</sup>—►



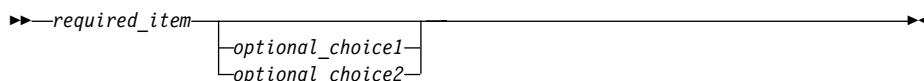
- Optional items appear below the main path.



- If you can choose from two or more items, they appear vertically, in a stack. If you *must* choose one of the items, one item of the stack appears on the main path.



If choosing one of the items is optional, the entire stack appears below the main path.



## Road map

This section identifies other sources of information about DB2 replication that you might find useful.

If you want to ...	Refer to ...
Learn about the IBM data replication solution	DB2 DataPropagator at <a href="http://www.ibm.com/software/data/dpropr/">www.ibm.com/software/data/dpropr/</a>  Data replication solution at <a href="http://www.ibm.com/software/data/dbtools/datarepl.html">www.ibm.com/software/data/dbtools/datarepl.html</a>  Database and data management at <a href="http://www.ibm.com/software/data/">www.ibm.com/software/data/</a>
Learn about last-minute changes to the product	The Installation Notes on the CD-ROM or the Release Notes that are installed with the products.  DB2 DataPropagator library at <a href="http://www.ibm.com/software/data/dpropr/library.html">www.ibm.com/software/data/dpropr/library.html</a>
Find technical support resources and customer support options	<a href="http://www.ibm.com/software/data/dpropr/support.html">www.ibm.com/software/data/dpropr/support.html</a>
Find classes available from IBM Learning Services	<a href="http://www.ibm.com/services/learning/">www.ibm.com/services/learning/</a>
Learn what's new this release in DB2 data replication	"What's new for DB2 replication for Version 8?" on page xvii.
Migrate from earlier versions of DB2 DataPropagator to Version 8	DB2 DataPropagator library at <a href="http://www.ibm.com/software/data/dpropr/library.html">www.ibm.com/software/data/dpropr/library.html</a>

<b>If you want to ...</b>	<b>Refer to ...</b>
Learn how to use the Replication Analyzer command-line tool to produce an HTML report about your replication environment.	DB2 DataPropagator library at <a href="http://www.ibm.com/software/data/dpropr/library.html">www.ibm.com/software/data/dpropr/library.html</a>
Read customer case studies	The Case studies page of the DataPropagator Web site at <a href="http://www.ibm.com/software/data/dpropr/casestudy.html">www.ibm.com/software/data/dpropr/casestudy.html</a>
Understand the Replication Center windows	The Replication Center help
Debug error messages	On OS/400: Type F1 when you receive an error message.  On UNIX, Windows, and z/OS: See Chapter 24, “Replication messages” on page 549.
Find out about other DB2 information	Product Web pages, which you will find at <a href="http://www.ibm.com/software/data/">www.ibm.com/software/data/</a> .

---

## How to send your comments

Your feedback helps IBM to provide quality information. Please send any comments that you have about this book or other DB2 DataPropagator documentation. You can use any of the following methods to provide comments:

- Send your comments from the Web. Visit the Web site at:  
[www.ibm.com/software/data/dpropr/](http://www.ibm.com/software/data/dpropr/)  
The Web site has a feedback page that you can use to enter and send comments.
- Send your comments by e-mail to [comments@vnet.ibm.com](mailto:comments@vnet.ibm.com). Be sure to include the name of the product, the version number of the product, and the name and part number of the book (if applicable). If you are commenting on specific text, please include the location of the text (for example, a chapter and section title, a table number, a page number, or a help topic title).

---

## What's new for DB2 replication for Version 8?

This section summarizes the major changes made to DB2 replication since Version 7. These changes include usability improvements, performance improvements, new function, serviceability improvements, changes to replication system commands, changes to control tables, and functions no longer supported. They are described in detail in the rest of this book.

---

### Usability improvements

***Enhanced handshake mechanism between the Capture and Apply programs:***

The handshake is a mechanism that the Apply program uses to tell the Capture program to start capturing data for a replication source. This mechanism has been changed and enhanced for Version 8. The Apply program inserts signals into the new signal (IBMSNAP\_SIGNAL) table to control when the Capture program should start capturing data for a source.

***Capture and Apply programs can be started in any order:*** In Version 8, you can start the Capture program after you start the Apply program, or you can start the Apply program after you start the Capture program. In Version 7, you had to start the Capture program before you started the Apply program.

***Adding registrations and subscription sets while the Capture program is running:*** You can register new replication sources, update existing registrations, add new subscription sets, or update existing subscription sets without reinitializing the Capture program or stopping and restarting it.

***Greater control over what is captured for each registration:*** When you register a table for replication you can specify whether you want the Capture program to capture changes for a row whenever *any* column of the table changes or only when a *registered* column changes. In previous versions, you could control what was captured using a start-up parameter for the Capture program, which meant all tables were treated the same. The start-up parameter is not available in V8 because you can control what is captured for each registration.

***Greater control over recapturing data from replicas:*** When you register a source, you can specify if you want changes recaptured from some tables but not others. By default:

- Changes are not recaptured from replica tables and forwarded to other replica tables.

- Changes to master tables in update-anywhere replication are recaptured and sent to replica tables.

**One Windows service per program:** In Version 7, you could create only one Windows service to operate all of your Capture and Apply programs. Now you can create *separate* services for each Capture and Apply program, as well as the Replication Alert Monitor. You can use each service to start or stop replication. You can use either the Replication Center or new commands to create (**asnscri** command) or drop (**asnsdrop** command) a service for replication programs.

**ARM support for the Capture and Apply programs:** For the z/OS environment, the Capture and Apply programs, and the Replication Alert Monitor, are enabled for the MVS Automatic Restart Manager (ARM). The ARM is an MVS recovery function that can improve the availability of specific batch jobs or started tasks. When a job or task fails, or the system on which it is running fails, the ARM can restart the job or task without operator intervention. The ARM uses element names to identify the applications with which it works, and each ARM-enabled application uses a unique element name that it uses to communicate with the ARM. The element names for replication are: ASNTCxxxxxyyy for the Capture program, ASNTAxxxxxyyy for the Apply program, and ASNAMxxxxxyyy for the Replication Alert Monitor.

**Improved messages:** Existing messages were improved and new messages were added. The explanation and user response sections were updated.

---

## Performance improvements

**Fewer joins between replication tables:** In Version 8, joins have been eliminated in some situations. The Apply program does not need to join the CD and UOW table to populate user copy target tables under many circumstances. Also, the CD and UOW tables do not need to be joined for pruning.

**Capture pruning runs concurrently with reading the DB2 log (UNIX, Windows, z/OS):** The Capture program reads the DB2 log while it prunes tables; therefore, pruning does not affect capture latency. In Version 7, the Capture program performed these tasks serially, not concurrently. Also, in Version 8, the Capture program prunes the UOW table, CD tables, trace tables, as well as the new signal (IBMSNAP\_SIGNAL) table and monitor (IBMSNAP\_CAPMON) table.

**Faster full refreshes of target tables (UNIX, Windows, z/OS):** DB2 replication takes advantage of the load improvements in the following DB2 products to provide faster full-refreshes of target tables:

- DB2 Universal Database for Windows and UNIX, Version 8

- DB2 Universal Database for z/OS and OS/390, Version 7 or later

***Apply program optimizes processing if it has only one subscription set:*** In Version 8, you can start the Apply program so that it will cache and reuse information about a single subscription set. Using the new **opt4one** keyword improves CPU usage and throughput rates.

***Fewer updates for subscription sets with multiple members:*** Compared to previous versions of DB2 replication, in Version 8 the Apply program makes fewer updates to control tables for subscription sets with multiple members.

---

## New user interface

With Version 8, you can set up and maintain your replication environment, operate the Capture and Apply programs, and the Replication Alert Monitor using one administration tool. The new DB2 Replication Center is a graphical tool that supports administration for DB2-to-DB2 replication environments and administration for replication between DB2 and non-DB2 relational databases.

The Replication Center is part of the DB2 Control Center set of tools and has the look and feel of the other DB2 centers. The Replication Center includes all of the replication function previously available from the DB2 Control Center and the DB2 DataJoiner Replication Administration (DJRA) tool. The Replication Center also has a launchpad that helps you perform the basic functions needed to set up a DB2 replication environment. The launchpad shows you graphically how the different steps are related to one another.

You can use the Replication Center to:

- Define defaults in profiles for creating control tables, source objects, and target objects
- Create replication control tables
- Register replication sources
- Create subscription sets and add subscription-set members to subscription sets
- Operate the Capture program
- Operate the Apply program
- Monitor the replication process
- Perform basic troubleshooting for replication

You can also use the Replication Center to perform many other replication administration tasks.

---

## New function

***Multiple Capture programs can concurrently read from the same DB2 log or journal:*** You can run more than one Capture program against a single DB2 log (DB2 catalog) or journal. For z/OS data-sharing groups, multiple Capture programs can read from the logs for the data-sharing group. Each Capture program is independent of any others. If necessary, you can register a single source table to more than one Capture program. Therefore, if you have low latency tables, they can have a dedicated Capture program so that they have different run-time priority and different Capture characteristics (such as pruning interval). Or, different organizations can maintain their own replication environments using the same source data, but different Capture programs. On z/OS platforms, you can use multiple Capture programs to support a mixture of ASCII, EBCDIC, and UNICODE source tables within a single DB2 subsystem.

***Multiple non-DB2 relational sources per federated database:*** If your replication environment includes non-DB2 sources, you can define multiple non-DB2 relational sources in a single federated database.

***Automated monitoring:*** The new Replication Alert Monitor runs continuously and monitors the Capture and Apply programs for you. You define thresholds for criteria that you want to monitor, and specify people who should be contacted automatically via e-mail when those thresholds are met or exceeded. You can use the Replication Center or two new commands (**asnmon** and **asnmcmd**) to configure and operate the Replication Alert Monitor.

***On-demand monitoring:*** You can query the status of the Capture, Apply, and Monitor programs using the **asnccmd**, **asnacmd**, **asnmcmd** status commands.

***Encrypted password file (UNIX, Windows):*** In Version 7, the password file used by the Apply program and the Replication Analyzer contained plain text, not encrypted information. In Version 8, the passwords in the password file are encrypted. No passwords are stored in plain text. A new command (**asnpwd**) enables you to create and maintain the password file.

***Improved ASNLOAD exit routine (UNIX, Windows, z/OS):*** The ASNLOAD exit routine is shipped as a sample exit routine in both source format (C) and compiled format. The sample exit routine differs on each DB2 platform, in each case taking advantage of the utility options offered on the platform. You can use the sample compiled program exit routine as provided and you can influence the behavior in some cases by customizing the replication configuration, or you can customize the exit routine code itself.

**More control over cold starts (UNIX, Windows, and z/OS) :** The warm start-up parameter is replaced by two parameters to give you more control over cold starts:

**warmsi**

If warm-start information is available, the Capture program resumes processing where it ended in its previous run; however, if this is the first time that the Capture program is starting or the new restart (IBMSNAP\_RESTART) table is empty, the Capture program switches to cold start. This is the default start-up parameter in Version 8.

**warmsa**

If warm-start information is available, the Capture program resumes processing where it ended in its previous run; however, if the Capture program cannot warm start, it switches to a cold start.

**More frequent commits by the Apply program:** In many situations, if you have user-copy, point-in-time, or replica target tables in a subscription set, you can specify that you want the Apply program to commit its work after it processes a specified number of transactions. To do so, you must run the Apply program in transaction mode.

**Referential integrity for more types of target tables:** In many situations, you can have referential integrity on user-copy and point-in-time target tables by starting the Apply program so that it commits its work in transaction mode.

**More ways to set operational parameters for the Capture program:** You can use the shipped defaults to operate the Capture program or you can create new defaults using the Capture parameters (IBMSNAP\_CAPPARMS) table to suit your replication environment. You can also supply operational parameters for the Capture program when you start the program, if you do not want to use the defaults for that session. While the Capture program is running, you can change the operational parameters using the Replication Center, the **chgparms** keyword of the **asncmd** command (UNIX, Windows, z/OS), or the **OVRDPRCAPA** command (iSeries). These changes last until you end the session or until you issue another change command.

**New option for replicating changes to target-key columns:** In Version 7, you could ensure that changes to key columns were replicated properly to your target tables by registering your source table to capture updates as delete/insert pairs. In Version 8, when you define a subscription set member, you can specify whether the Apply program should use the before-image values or the after-image values when the Apply program builds a WHERE clause using the primary-key columns in its predicates. By using before-image values, you can avoid the conversion of an update to an insert. You can

specify either that a registration use delete/insert pairs for updates or that the subscription-set member use before-image values in Apply WHERE-clause predicates.

***Additional historical data in control tables:*** DB2 replication provides additional historical data in the control tables, which describe replication activities. Two new tables that contain such data are the Apply trace (IBMSNAP\_APPLYTRACE) table and the Capture monitor (IBMSNAP\_CAPMON) table. You can query the data using the Replication Center.

***More tables pruned by the Capture program:*** The Capture program prunes the following tables: CD tables, UOW table, trace (IBMSNAP\_CAPTRACE) table, as well as the new signal (IBMSNAP\_SIGNAL) table and monitor (IBMSNAP\_CAPMON) table. Also, for OS/400 environments the Capture program prunes the Apply-qualifier cross-reference (IBMSNAP\_AUTHTKN) table.

***Longer table names and column names:*** DB2 replication now supports source table and target table names up to 128 characters, and column names up to 30 characters for databases that support long names.

***Adding columns to source and CD tables while the Capture program is running:*** You can add columns to your replication source tables without reinitializing the Capture program or stopping and restarting it. On UNIX, Windows, and z/OS, you can also alter the CD table while the Capture program is running.

***New signals to control the Capture program:*** The Capture program can now be controlled by signals written to the signal (IBMSNAP\_SIGNAL) table. The signal table provides a way to communicate with the Capture program through log records. Capture uses the signals for the following situations:

- To determine when to start capturing changes for a particular table
- To determine when to stop capturing changes for a particular table<sup>1</sup>
- To determine when to terminate
- Whether it must perform update-anywhere replication
- To provide the log sequence number for setting a precise end point for Apply events

Not only does the signal table let the Apply program tell the Capture program when to start capturing data, it also allows for precise termination of log record reading and for user-defined signals through log records.

---

1. This signal is not supported for OS/400 environments.



***Replicating Data Links values (AIX, Solaris Operating Environment, Windows, iSeries):***

- If you have a DATALINK value pointing to an external file, you can retrieve consistent versions of files if the column is defined with RECOVERY YES. In past releases, DB2 would replicate the latest copy of the file and could not guarantee that the file being replicated was consistent with the replicated database data values.
- You can maintain the same target file across multiple changes in the source database.
- For the AIX and Windows operating systems, and the Solaris operating environment, you can connect to the DB2 Data Links Manager replication daemon (DLFM\_ASNCOPYD) to retrieve and store Data Links files for replication. You do not need to start and maintain a separate ASNDLCOPYD daemon as in previous releases. On OS/400, you still need to start and maintain a separate ASNDLCOPYD daemon.

***Unicode encoding schemes added (z/OS):*** DB2 DataPropagator for z/OS Version 8 supports UNICODE and ASCII encoding schemes. This function was introduced in DB2 DataPropagator for OS/390 Version 7.

***64-bit support added (Windows, UNIX, z/OS):*** In Version 8, you can replicate on platforms where DB2 offers 64-bit support. Applications running on 64-bit operating systems benefit from the increased memory address space that these systems provide.

***Migration utility:*** The new replication migration utility (**asnmig8**) consists of a set of migration scripts that you can use to convert all Version 5, Version 6, or Version 7 replication tables to Version 8 formats.

---

## **Serviceability improvements**

***New trace facility (UNIX, Windows, z/OS):*** The new replication trace facility (**asntrc**) is similar to the DB2 trace facilities. You can start or stop the trace facility without stopping and restarting the Capture and Apply programs. Also, the trace output is compact, which usually results in smaller trace files than were generated in previous releases, and is consistent with the DB2 trace format.

***Improved tracing (iSeries):*** New tracing points were added to the Version 7 trace function to provide more debugging information.

***Replication Analyzer program updated:*** The Replication Analyzer program was modified to analyze the new V8 features. The Analyzer generates reports about the state of the replication control tables on the specified systems. These

reports can be used to verify and tune your replication environment or to diagnose problems. You can download the Analyzer and its documentation from the Web.

---

## Changes to replication system commands

### *New and changed replication system commands for UNIX, Windows, z/OS:*

The syntax of existing system commands on Windows, UNIX, and z/OS was modified. The following changes were also made:

- The Capture command line (**asncmd**) was renamed to **asnccmd** so that it is consistent with the new Apply command line (**asnacmd**), which you use to operate the Apply program, and the new Monitor command line (**asnmcmd**), which you use to operate the Monitor program.
- The **asnccp** command for starting the Capture program was renamed to **asnccap**.

The following new system commands, which run on UNIX, Windows, and z/OS operating systems, were added:

- **asnacmd** (Apply command line) operates and stops the Apply program.
- **asnmon** (Monitor command) starts the Replication Alert Monitor
- **asnmcmd** (Monitoring command line) operates and stops the Replication Alert Monitor.
- **asnanalyze** (Analyzer command) generates reports about the state of the replication control tables.
- **asnpwd** (Password command) creates and maintains password files needed in a distributed replication environment.
- **asntrc** (Trace facility) replaces the startup options to generate a trace for the Capture and Apply programs.

### *New and changed replication system commands for OS/400 operating systems (iSeries):*

The following new system commands, which run on an OS/400 system, were added:

- **ADDDPRREG** (Add a DPR registration) registers a user table for replication.
- **RMVDPRREG** (Remove a DPR registration) removes a user table from the list of source tables available for replication.
- **ADDDPRSUB** (Add DPR subscription set) creates an empty subscription set or a subscription set with one member.
- **RMVDPRSUB** (Remove DPR subscription set) removes an empty set or a set and all of its members.
- **ADDDPRSUBM** (Add DPR subscription-set member) adds a member to an existing subscription set.

- **RMVDPRSUBM** (Remove DPR subscription-set member) removes a single subscription-set member from a subscription set.
- **OVRDPRCAPA** (Override DPR Capture attributes) changes the attributes for the Capture program that is currently running.
- **ANZDPR** (Analyzer) generates reports about the state of the replication control tables on the specified systems. These reports can be used to verify and tune your replication environment or to diagnose problems.
- **WRKDPRTRC** (Trace options) operates various trace options such as Dump.

Some changes were made to existing system commands for OS/400 systems:

- **DPRVSN** (DataPropagator Version) parameter was removed from all system commands.
- **CAPCTLLIB** (Capture Control Library) parameter was added to the Capture commands.
- New parameters were added to the **CHGDPRCAPA** (Change DPR Capture attributes) and **STRDPRCAP** (Start DPR Capture) commands to take advantage of the new tracing and monitoring functions.
- A new parameter was added to the **ENDDPRCAP** (End DPR Capture) command so that it automatically reorganizes the CD and UOW tables to reclaim space.
- New parameters were added to the **STRDPRAPY** (Start DPR Apply) command that enable the Apply program to run only once, clean up the Apply trail (IBMSNAP\_APPLYTRAIL) table, and optimize processing of single subscription sets.

---

## Changes to control tables

Substantial changes were made to the control table structures in Version 8 to support new function and to improve usability. New tables were added, some existing tables were changed, and a few tables were made obsolete by new tables.

### *The following new tables were added:*

- **IBMSNAP\_APPENQ** ensures that only one Apply program is running for a single Apply qualifier.
- **IBMSNAP\_APPLYTRACE** contains important messages from the Apply program.
- **IBMSNAP\_CAPENQ** ensures that only one Capture program is running for a single Capture schema.

---

2. This table is not used on OS/400.

- IBMSNAP\_CAPMON contains operational statistics for monitoring the progress of the Capture program.
- IBMSNAP\_CAPSCHEMAS contains the names of all Capture schemas.
- IBMSNAP\_PRUNE\_SET coordinates the pruning of CD tables.
- IBMSNAP\_RESTART enables the Capture program to resume capturing from the correct point in the log or journal.
- IBMSNAP\_SIGNAL contains signals used to control the Capture program.

***The following new tables were added for the Replication Alert Monitor:***

- IBMSNAP\_ALERTS contains a history of all alerts issued by the Replication Alert Monitor.
- IBMSNAP\_CONDITIONS contains alert conditions for each monitored server.
- IBMSNAP\_CONTACTGRP maps contacts with groups.
- IBMSNAP\_CONTACTS contains contact names and addresses.
- IBMSNAP\_GROUPS contains contact groups.
- IBMSNAP\_MONENQ ensures that only one Monitor process is running for a single Monitor qualifier.
- IBMSNAP\_MONSERVERS contains the most recent time that the Replication Alert Monitor monitored a Capture or Apply control server.
- IBMSNAP\_MONTRACE traces Replication Alert Monitor activity.
- IBMSNAP\_MONTRAIL contains a history of Monitor activity for every Monitor cycle.

***The following tables were changed:***

- IBMSNAP\_APPLYTRAIL
- IBMSNAP\_AUTHTKN (OS/400 only)
- IBMSNAP\_CAPPARMS (formerly known as IBMSNAP\_CCPPARMS)
- IBMSNAP\_CAPTRACE (formerly known as IBMSNAP\_TRACE)
- IBMSNAP\_PRUNCNTL
- IBMSNAP\_REG\_EXT (OS/400 only)
- IBMSNAP\_REGISTER
- IBMSNAP\_SUBS\_COLS
- IBMSNAP\_SUBS\_EVENT
- IBMSNAP\_SUBS\_MEMBR
- IBMSNAP\_SUBS\_SET
- IBMSNAP\_UOW

CD tables were also changed.

***The following tables from previous versions of DB2 replication are now obsolete:***

- IBMSNAP\_CRITSEC is replaced by IBMSNAP\_SIGNAL.
- IBMSNAP\_WARMSTART is replaced by IBMSNAP\_RESTART.

---

## **Functions no longer supported**

The DB2 DataJoiner Replication Administration (DJRA) tool is not supported for Version 8. You cannot use DJRA to create Version 8 replication control tables, and you cannot use DJRA to register sources or define subscription sets that use V8 control tables. DJRA continues to be supported for Version 7 replication environments. Use the Replication Center for V8 replication environments.

The DB2 Control Center does not support Version 8 replication control tables, and you cannot use the Control Center to register sources or define subscription sets that use V8 control tables. You can use the Control Center for Version 7 replication environments. Use the Replication Center for V8 replication environments.

The **ASNSAT** command is no longer available. Also, the ability to generalize replication subscriptions and set up a DB2 satellite replication environment is no longer available from the Satellite Administration Center. If you require data replication for a mobile work force, consider migrating your satellite DB2 databases to DB2 Everyplace, Version 8. For additional information, contact your IBM representative.



---

## Part 1. Replication guide

This part of the book contains the following chapters:

Chapter 1, “Planning for replication” on page 3 describes how to plan your replication environment.

Chapter 2, “Setting up for replication” on page 15 describes how to prepare your environment for replication.

Chapter 3, “Registering tables and views as replication sources” on page 37 describes what you need to know to register replication sources.

Chapter 4, “Subscribing to sources” on page 63 describes what you need to know to create subscription sets and add members to subscription sets.

Chapter 5, “Replicating special data types” on page 97 describes the replication options for LOB and DATALINK values in source tables.

Chapter 6, “Subsetting data in your replication environment” on page 107 describes how to customize what data is captured and applied to the target as well as how the data is applied to the target.

Chapter 7, “Manipulating data in your replication environment” on page 111 describes how to use the Capture program or the Apply program to manipulate the source data.

Chapter 8, “Customizing and running replication SQL scripts” on page 115 describes how to run SQL in your replication environment.

Chapter 9, “Operating the Capture program” on page 117 describes how to operate the Capture program for all operating-system environments.

Chapter 10, “Operating the Apply program” on page 139 describes how to operate the Apply program for all operating-system environments.

Chapter 11, “Monitoring replication” on page 161 describes how to monitor your replication environment.

Chapter 12, “Making changes to your replication environment” on page 183 describes how to change your replication environment.

Chapter 13, “Maintaining your replication environment” on page 225 describes how to maintain your source tables, control tables, and target tables.



---

## Chapter 1. Planning for replication

This chapter describes how to plan your replication environment. It contains the following sections:

- “Memory planning”
- “Storage planning” on page 5
- “Planning for conflict detection” on page 11
- “Planning for non-DB2 relational sources” on page 12
- “Planning for code page translation” on page 13

---

### Memory planning

You must plan for the amount of memory required by DB2 replication. DB2 replication uses memory only as needed. The amount of memory required is directly proportional to how much data is being replicated from the source and the concurrency of the transactions. Basically, the more data that is being replicated and the more concurrent transactions you have, the more memory is required by replication.

Running the Capture and Apply programs can consume a significant amount of memory resources.

#### Memory used by the Capture program

When the Capture program reads the DB2 log, the Capture program stores individual transaction records in memory until it reads the associated commit or abort record. Data associated with an aborted transaction is cleared from memory, and data associated with a commit record is written to the CD table and the UOW table. The committed transactions stay in memory until the Capture program commits its work when it reaches its commit interval.

To monitor how much memory the Capture program is using, look in the `CURRENT_MEMORY` column of the Capture monitor (`IBMSNAP_CAPMON`) table.

You can set the **memory\_limit** parameter when you start the Capture program to ensure that Capture uses a specified amount of memory for storage that is associated with transactions. Other storage use is not limited by this parameter. You can also change the **memory\_limit** parameter while the Capture program is running. If Capture reaches the memory limit, it writes some transactions to a spill file. See “Planning space requirements for spill files for the Capture program” on page 10 for the storage requirements of spill

files. You need to consider the memory resources that are used by the Capture program in relation to its storage space requirements.

You should also consider the size of user transactions and the commit interval when planning for the Capture program's memory requirements. Long running batch jobs without interim commits take a lot of memory when you run the Capture program. Generally, the smaller the commit interval, the less memory required by the Capture program.

**Reading information about registrations:** Information about active registrations is read and stored in memory when the Capture program is instantiated and when you add registrations dynamically while the Capture program is running.

**Reading log records (UNIX, Windows, z/OS):** When DB2 replication reads log records it uses a memory buffer. The default size of the buffer on UNIX and Windows operating systems is fifty 4 KB pages. The default size on the z/OS operating system is sixty-six 1 KB pages, and it is ECSA (extended common service area) storage. Replication uses ECSA only in this situation.

**Memory used on OS/400:** CURRENT\_MEMORY is the up-to-date account of extra memory allocated for holding the transaction records beyond the memory used by standard I/O buffers for the active CD tables. It is an indication of how much extra memory is being used to hold the large number of transactions. It is not an accurate sum of all the memory used by the specific journal job.

Information stored in the Capture monitor (IBMSNAP\_CAPMON) table provides operational statistics to help you tune memory usage. Note that the values in this table are for a particular Capture monitor interval, they are not cumulative across monitor intervals. The data in the CURRENT\_MEMORY column does not contain an additive count. It reflects the memory in use at the end of the monitor interval when the record is created. The Capture monitor interval determines how frequently the Capture program inserts data into this table. Use one of the following methods to tune the amount of memory being used by the Capture program:

**Tuning memory limit to allow for spills:**

1. When you start the Capture program, use the default memory limit.
2. Check if data spilled from memory to a temporary file by looking at the TRANS\_SPILLED column in the Capture monitor (IBMSNAP\_CAPMON) table. This column shows the number of source system transactions that spilled to disk due to memory restrictions during a particular Capture monitor interval.

3. If data spilled from memory, either use a higher memory limit or a lower commit interval.

**Tuning memory limit to prevent spills:**

1. When you start the Capture program, set a high memory limit. (How high depends on your system resources.)
2. Check how much memory is being used by looking at the CURRENT\_MEMORY column in the Capture monitor (IBMSNAP\_CAPMON) table. This column shows the amount of memory (in megabytes) that the Capture program used during a particular Capture monitor interval.
3. If much less memory is being used than what you specified for the memory limit, set a lower value for the memory limit.

**Memory used by the Apply program**

When the Apply program fetches data, it typically uses a small amount of memory for fetching individual rows. The amount of memory used is proportional to the size of the table columns and the number of rows fetched at one time. For example, if the Apply program is fetching a LOB column, it could potentially use 2 GB of memory.

Information about active subscription sets is read and stored in memory when the Apply program is running. The amount of memory used at one time by the Apply program is generally proportional to the amount of memory required to process the subscription set that has the most members.

**Memory used by the Replication Alert Monitor**

Memory is used for storing the definitions and for keeping the alerts in memory before they are sent as notifications. The amount of memory needed for the definitions is directly proportional to the number of definitions. The Replication Alert Monitor reserves 32 KB of memory for storing alert notifications. More memory is requested, as needed, and released when no longer required.

---

**Storage planning**

In addition to the storage required for DB2, you must ensure that storage is available for replication for the following items:

**Database log and journal data**

The additional data logged to support the replication of data. See “Planning log impact” on page 6 for details.

**Target tables and control tables**

The replicated data and control tables (including CD tables). See “Planning the storage requirements of target tables and control tables” on page 8 for details.

### **Temporary files**

The data stored by replication programs in spill files and diagnostic log files (for example, \*CAP.log and \*APP.log). See “Planning storage requirements for temporary files” on page 9 for details.

### **OS/400: Current receiver size for Capture**

For registered source tables yet to be captured, the journal entries must remain on the current chain of receivers. For more information, see “Using the delete journal receiver exit routine” on page 35.

All of the sizes given in the following sections are estimates only. To prepare and design a production-ready system, you must also account for such things as failure prevention. For example, the holding period of data (discussed in “Planning the storage requirements of target tables and control tables” on page 8) might need to be increased to account for potential network outages.

**Tip:** If storage estimates seem unreasonably high, reexamine how frequently the Apply program runs subscription sets and how frequently your replication tables are pruned. You must consider trade-offs between storage usage, capacity for failure tolerance, and CPU overhead.

## **Planning log impact**

You must plan the log impact for the replication servers. DB2 replication requires that both the source and the target tables be logged (journaled).

### **Planning the log impact for DB2 source servers**

In general you need an additional three times the current log volume for all tables involved in replication. Basically, you need log space for the source table as well as the CD table and the replication control tables. This section provides other factors that can help you make a more accurate estimate of the log impact that you can expect in your replication environment.

Consider the updates made to the source database by your applications and the replication requirements. For example, if an updating application typically updates 60% of the columns in a table, the replication requirements could cause the log records to grow by more than half compared to a similar table that is not replicated.

### **UNIX, Windows, and z/OS:**

- DB2 logs full-row images for each UPDATE statement. This occurs because, before you can replicate a table, you must create it (or alter it) with the DATA CAPTURE CHANGES keywords.
- One of the replication requirements that adds the most to the log is the capturing of before- and after-images (as for replica target tables in update-anywhere replication scenarios). One way to reduce the log volume

is to reduce the number of columns defined for the replication source. For example, do not capture before-images if they're not required.

#### **OS/400:**

- DB2 logs full-row images for each UPDATE statement. One way to reduce the log volume is to reduce the number of columns defined for the replication source, for example, do not capture before-images if they're not required.
- To minimize the amount of storage used for CD tables and UOW tables, frequently reorganize these tables because pruning does not recover DASD for you. You can use the keyword RGZCTLTBL (Reorganize control tables) on the **ENDDPRCAP** command to reorganize control tables. Observe the DASD usage patterns under normal operating conditions to help you predict and manage DASD usage. If journaling is on, also take into account that the log or journal volume increases as DB2 log insertions to and deletions from the UOW table and CD tables.
- When the current receiver is full, the system switches to a new one; you can optionally save and delete old ones no longer needed for replication. When a system handles a large number of transactions, the Capture program can occasionally lag behind. If Capture is frequently lagging behind, you can separate your source tables into multiple journals to distribute the workload to multiple instances of the Capture program.

#### **Planning the log impact for target servers**

In addition to logging for the source database, there is also logging for the target database, where the rows are applied. The impact to the log depends on the commit mode that you choose for the Apply program.

##### **Table mode**

In table-mode processing, the Apply program issues a single commit after all fetched data is applied. The Apply program does not issue interim checkpoints. In this case, you should estimate the maximum amount of data that the Apply program will process in one time interval and adjust the log space to accommodate that amount of data.

##### **Transaction mode**

In transaction-mode processing, the Apply program copies every update in the source transaction order to the target tables and commits these changes on a transaction boundary at an interval. You set the interval for the interim commits by setting the value of *x* in the subscription set option **commit\_count(x)**. After the Apply program fetches all answer sets, it applies the contents of the spill files in the order of commit sequence. This type of processing allows all spill files to be open and processed at the same time. For example, if you set

commit count to 1, the Apply program commits after each transaction, if you set commit count to 2, it commits after each set of two transactions.

**OS/400:** If the target operating system is OS/400, you also need to consider the log space (journal receivers space) of the target tables. Because journal receivers for target tables on OS/400 can be created with the MNGRCV(\*SYSTEM) and DLTRCV(\*YES) parameters, and because you need to journal only the after-image columns, use the following formula to estimate the volume of the journal receivers for the target tables:

`journal_receiver_volume=target_table_row_length X journal_receiver_threshold`

## **Planning the storage requirements of target tables and control tables**

You must estimate the volume of new target tables. The space required for a target table is usually no greater than that of the source table, but can be much larger if the target table is denormalized or includes before-images (in addition to after-images) or history data. Target table size depends on what you choose to replicate, for example, the percentage of the source table you are replicating, the data type of columns you're replicating, whether you're replicating before- and after-images, whether you're adding computed columns, whether you're subsetting rows, whether any transformations are performed during replication.

The CD tables and some replication control tables (IBMSNAP\_UOW, IBMSNAP\_CAPTRACE, IBMSNAP\_APPLYTRACE, IBMSNAP\_APPLYTRAIL, IBMSNAP\_CAPMON, IBMSNAP\_ALERTS) also affect the disk space required for DB2 source databases. These tables can grow very large depending on how you set up your replication environment. The space required for the other replication control tables is generally small and static.

The CD tables grow in size according to the amount of data replicated until the Capture program prunes them. To save space, you might want to define your CD tables in compressed table spaces. To estimate the space required for the CD tables, first determine how long you want to keep the data before pruning it, then specify how often the Capture program should automatically prune these tables or how often you will prune the tables using a command.

When calculating the number of bytes of data replicated, you need to include 21 bytes for overhead data that is added to the CD tables by the Capture program. When calculating the number of bytes of data replicated, you need to include 21 bytes for overhead data for each row that is added to the CD tables by the Capture program. Determine the period of time for which the Capture program should be able to keep capturing data into CD tables, even when the data cannot be applied - for example, in the case of a network outage. Estimate the number of inserts, updates, and deletes that typically would be captured for the source table within that contingency time period.

To determine the recommended size for the CD table, use the following guideline:

```
recommended_CD_size =  
    ( (21 bytes) + sum(length of all registered columns) ) X  
    (number of inserts, updates, and deletes to source table  
    during the contingency period)
```

**Example:** If the rows in the CD table are 100 bytes long (plus the 21 bytes for overhead), and 100,000 updates are captured during a 24-hour contingency period, the storage required for the CD table is about 12 MB.

Registered columns in this formula include both before- and after-image columns. If updates are being converted to pairs of INSERT and DELETE operations, then take them into account when determining the total number of inserts, updates, and deletes. For example, count each update to the source table as two rows in the CD table.

The UOW table grows and shrinks based on the number of rows inserted by the Capture program during a particular commit interval and on the number of rows that are pruned. A row is inserted in the UOW table each time an application transaction issues a COMMIT and the transaction executed an INSERT, DELETE, or UPDATE operation against a registered replication source table. You should initially over-estimate the space required by the table and monitor the space actually used to determine if any space can be recovered.

## Planning storage requirements for temporary files

You must plan for the storage requirements of spill files and diagnostic log files.

### Planning space requirements for diagnostic log files (UNIX, Windows, z/OS)

Diagnostic log files store information about the activities of Replication programs, such as when the program started and stopped, and other informational or error messages from the program. By default, the program appends messages to its log file, even after the program is restarted. Ensure that the directories that contain these log files have enough space to store the files. The location of these files depends on the value that you set for the **capture\_path**, **apply\_path**, and **monitor\_path** start-up parameters when you started the Capture program, Apply program, and Replication Alert monitor program, respectively.

If you are concerned about storage, you have the option of reusing the program logs so that each time the program starts it deletes its log and recreates it. You can specify if you want to reuse the log when you start the program.

### **Planning space requirements for spill files for the Capture program**

If the Capture program does not have sufficient memory, it writes (or spills) transactions to spill files. The Capture program writes the biggest transaction to file; however, the biggest transaction is not necessarily the one that exceeded the memory limit.

- **UNIX, Windows:** On UNIX and Windows, spill files are always on disk. One file per transaction is created in the **capture\_path** directory.
- **OS/400:** On OS/400, spill files are created in library QTEMP, one spill file for each registration that needs a spill file.
- **z/OS:** On z/OS, spill files go to VIO.

The size of the Capture spill files depends on the following factors:

#### **Memory limit**

Use the **memory\_limit** operational parameter to specify how much memory can be used by the Capture program. The more memory you allow, the less likely the Capture program will spill to files.

#### **Size of transactions**

Larger transactions might increase the need to spill to file.

#### **Number of concurrent transactions**

If the Capture program processes more transactions at the same time, or processes interleaved transactions, the Capture program needs to store more information in memory or on disk.

#### **Commit interval**

Typically the lower the commit interval the lower the need for storage because Capture has to store information in memory for a shorter period of time before committing it.

### **Planning space requirements for spill files for the Apply program**

The Apply program requires temporary space to store data. (If you are using the ASNLOAD utility, you might have a load input file instead of a load spill file.) The Apply program uses spill files to hold the updates until it applies them to the target tables. In general, the spill files are disk files; however, on z/OS platforms, you can specify that data be spilled to memory. Unless you have virtual memory constraints, store the spill files in virtual memory rather than on disk.

The size of the spill file is proportional to the size of the data selected for replication during each replication interval. Typically the spill file is approximately two times the size of the data. You can estimate the size of the spill file by comparing the frequency interval (or data-blocking value) planned for the Apply program with the volume of changes in that same time period (or in a peak period of change).

On OS/400, the spill file's row size is a constant 32 KB.



On UNIX, Windows, z/OS, the spill file's row size is the *target row size*, including any replication overhead columns. The row size is not in DB2 packed internal format, but is in expanded, interpreted character format (as fetched from the SELECT). The row also includes a row length and null terminators on individual column strings. The following example estimates the size of the spill file that is required for the data selected for replication and it does not take into account the extra space needed for the other data that is stored in the spill file.

**Example:** If change volume peaks at 12,000 updates per hour and the Apply program frequency is planned for one-hour intervals, the spill file must hold one-hour's worth of updates, or 12,000 updates. If each update represents 100 bytes of data, the spill file will be approximately 1.2 MB at a minimum. Additional space is required for the other data that is stored in the spill file.

---

## Planning for conflict detection

If you use standard or enhanced conflict detection, you must store before-images in the CD (or CCD) tables for the replica target tables. Also, the referential integrity rules are restricted. In peer-to-peer and update-anywhere scenarios, or when the Apply program uses transaction mode processing, you should define referential integrity rules that are in keeping with the source rules.

If you use peer-to-peer replication or update-anywhere replication and you do not want to turn on conflict detection, you should design your application environment to prevent update conflicts. If conflicts cannot occur in your application environment, you can save processing cycles by not using conflict detection.

Use either of the following methods to prevent conflicts in peer-to-peer and update-anywhere replication:

### Fragmentation by key

Design your application so that the replication source is updated by replicas for key ranges at specific sites. For example, your New York site can update sales records only for the Eastern United States (using ZIP codes<sup>3</sup> less than or equal to 49999 as the key range), but can read all sales records.

### Fragmentation by time

Design your application so that the table can be updated *only* during specific time periods at specific sites. The time periods must be sufficiently separated to allow for the replication of any pending

---

3. United States postal codes.

changes to be made to the site that is now becoming the master version. Remember to allow for time changes, such as Daylight Savings Time or Summer Time, and for time-zone differences.

---

## **Planning for non-DB2 relational sources**

Capture triggers are used instead of the Capture program if you are replicating from non-DB2 relational databases. These triggers capture changed data from a non-DB2 relational source table and commit the changed data into CCD tables. Capture triggers affect your transaction throughput rates and log space requirements. Also, if you have existing triggers in your environment you might need to merge them with the new Capture triggers. For more information, see the following sections:

- “Planning transaction throughput rates for Capture triggers”
- “Planning the log impact for non-DB2 relational source servers”
- “Planning locks for Oracle source servers”
- “Planning coexistence of pre-existing triggers with Capture triggers” on page 13

### **Planning transaction throughput rates for Capture triggers**

The transaction workload for your source system will increase; trigger-based change capture has an impact on transaction throughput rates. Capture triggers also increase the response time for the updating transactions. The impact is greatest for those transactions that heavily update application source tables that are to be replicated.

### **Planning the log impact for non-DB2 relational source servers**

For non-DB2 relational source servers, your source applications will need more active log space because the log volume approximately triples for replicated source tables. Changes are captured by triggers on the source tables and are stored in CCD tables, changed data is written within the same commit scope as the changing source tables, and data is later deleted through a trigger-based pruning mechanism. Therefore, each source INSERT, UPDATE, or DELETE operation becomes an INSERT, UPDATE, or DELETE operation, plus an INSERT operation, plus a DELETE operation. The log volume increases even more if you change updates to pairs of DELETE and INSERT operations.

If you run out of log space and the Capture trigger cannot insert a record into the CCD table, the transaction attempted by the user or application program will not complete successfully.

### **Planning locks for Oracle source servers**

Any application currently updating the Oracle source must finish before the Apply program can start applying data. The Apply program must lock the CCD table so that it can process data and set its synchpoint. The locks on the

CCD tables are held only until the Apply program sets its synchpoint, not through the entire Apply cycle. Applications that need to update the source table must wait until the Apply program unlocks the CCD table.

### **Planning coexistence of pre-existing triggers with Capture triggers**

The Capture trigger logic is in the SQL script generated by the Replication Center when you register a source. By default, an INSERT trigger, an UPDATE trigger, and a DELETE trigger are created so that those types of changes (insert, update, delete) can be replicated from the source table. The trigger name consists of the name of the CCD table preceded by a letter describing the type of trigger: I for INSERT, U for UPDATE, D for DELETE. For example, if the CCD table name is undjr02.ccd001, the name of the generated DELETE trigger is undjr02.dccd001. You must *not* change the names of the triggers that are generated in the script.

If a trigger already exists on the table that you want to register for replication and that trigger has the same name as the one that is in the generated script, you'll receive a warning when the script is generated. Do *not* run the generated script because the RDBMS might overwrite the existing trigger. Determine how you want to merge the pre-existing triggers with the new triggers, and create a script that merges your existing logic with the trigger logic generated by the Replication Center.

If the type of trigger that you want to create already exists on the table that you want to register for replication, and the RDBMS allows only one such trigger per table, you must merge the logic before you run the generated script.

---

## **Planning for code page translation**

Replication components are database applications that rely on the DB2 databases on various platforms to handle code page translation of data. They work with data using SQL SELECT, INSERT, UPDATE, and DELETE statements.

### **Replicating data between databases with compatible code pages**

If your replication configuration requires SQL statements and data to go between systems with differing code pages, the underlying DB2 protocols such as DRDA handle code page translation. Also, if data is passed between DB2 and non-DB2 relational databases, DB2 replication relies on the underlying database products to handle any necessary code page translation.

If you plan to replicate data between databases with differing code pages, check the *DB2 Administration Guide* to determine if the code pages you have are compatible. For example, if you are using DB2 for UNIX or Windows see the section on the conversion of character data.

Once you have verified that your databases have compatible code pages, determine if the databases use code pages differently. For example, assume that one database product allows a different code page for each column in a table while another database product does not allow different code pages per column, it requires the code page to be specified only at the database level. A table with multiple code pages in the first product cannot be replicated to a single database in the second product. Therefore, how the databases handle code pages affects how you must set up replication to ensure that data is successfully replicated between the various databases in your environment.

## **Configuring National Language Support (NLS) for Replication**

The NLS configuration for replication is defined when you set up database connectivity between systems. However, if you are running the Capture program on UNIX or Windows platforms, the Capture program must use the same code page as the database from which it is capturing the data. If it does not use the same code page, you must set a DB2 environment variable or registry variable called DB2CODEPAGE.

DB2 derives the code page for an application from the active environment in which the application is running. Typically, when DB2CODEPAGE is not set, the code page is derived from the language ID, as specified by the operating system. In most situations, this value is correct for the Capture program if you use the default code page when you create your database. However, if you create your database with an explicit code page that is something other than the default code page, you must set the DB2CODEPAGE variable for the Capture program. Otherwise, data might not be translated correctly when the Capture program inserts it into a CD table. The value that you use for the DB2CODEPAGE variable should be the same as what you specify on your CREATE DATABASE statement. Refer to the *DB2 Administration Guide* for information about setting the DB2CODEPAGE variable.

---

## Chapter 2. Setting up for replication

You must set up your environment before you can replicate data.

This chapter contains the following sections:

- “Controlling access to replication servers”
- “Authorizing user IDs for replication” on page 17
- “Storing user IDs and passwords for replication (UNIX, Windows)” on page 23
- “Setting up the replication control tables” on page 23
- “Setting up the replication programs” on page 26
- “Setting up journals (OS/400)” on page 32

---

### Controlling access to replication servers

In most replication environments, data is distributed across servers. If you have such an environment, you must ensure that the replication programs can connect to all servers. You must have the correct software installed to provide connectivity between servers, and you must configure the connectivity between the servers. If you are replicating to non-DB2 relational databases, you must also configure the federated server and related connectivity.

### Connectivity requirements for replication

Any workstation that runs the Apply program, the Replication Center, or the replication commands must be able to connect to the source server, Capture control server, Apply control server, and target server databases.

If you use the Replication Alert Monitor, the workstation on which it runs must be able to connect to the Monitor control server and to any server that it monitors. If you want to use the Replication Center to set up monitoring, ensure that the Replication Center can connect to the Monitor control server.

If your replication design involves staging data at a server that is different from the source database, you must carefully consider the communications between the various servers. Be sure to limit the layers of emulation, LAN bridges, and router links required, because these can all affect replication performance.

When the databases are connected to a network, connectivity varies according to the platforms being connected.

## Connecting to non-DB2 relational servers

If you want to replicate data to or from a non-DB2 relational server, you must be able to access the non-DB2 relational server and connect to it.

Before you attempt to replicate from non-DB2 relational source servers, you must set up your federated server and database. There are three main setup steps:

1. Define a wrapper so that the DB2 database can access other non-DB2 relational databases.
2. Define a non-DB2 relational database using a server mapping.
3. If the user ID and password combination that is used to connect to the DB2 database differs from the one used to access the non-DB2 relational database, you must create a user mapping.

Follow the instructions in *DB2 Federated Systems Guide*, GC27-1224, to ensure that your environment is correctly configured.

## Connecting to z/OS or iSeries servers from UNIX or Windows servers

Ensure that you can connect to all remote servers. To configure connections between z/OS or OS/400 systems and Windows or UNIX systems, refer to the *DB2 Connect Quick Beginnings*.

### Procedure:

To connect to an iSeries server from a DB2 for Windows workstation:

1. Before you connect to an iSeries server from a DB2 for Windows client, make sure that your workstation is set up correctly:
    - You must have a DB2 Universal Database or DB2 Client Application Enabler (CAE) client installed on your workstation.
    - You must have TCP/IP set up on your workstation.
  2. Log on to the iSeries server and locate the relational database:
    - a. Log on to the iSeries server to which you want to connect.
    - b. Submit a **dsprdbdire** command, then specify `local` for `*LOCAL`.
    - c. Locate the name of the relational database in the output. For example, in the following output, the database is called DB2400E:
- |          |             |
|----------|-------------|
| MYDBOS2  | 9.112.14.67 |
| RCHASDPD | RCHASDPD    |
| DB2400E  | *LOCAL      |
| RCHASLJN | RCHASLJN    |
- a. Catalog the OS/400 database in DB2 for Windows:
    - a. From your Windows NT workstation, click **Start → Programs → IBM DB2 → Command Window**. The DB2 CLP command window opens.

- b. In the command window, type the following three commands in exact order:

```
db2 catalog tcpip node server_name remote server_name server 446 system
server_name ostype OS400
```

```
db2 catalog dcs database rdb_name AS rdb_name
```

```
db2 catalog database rdb_name AS rdb_name at node server_name
authentication dcs
```

Where *server\_name* is the TCP/IP host name of the iSeries system, and *rdb\_name* is the name of the iSeries relational database that you found in Step 2 on page 16.

4. In the command window, issue the following command:

```
db2 terminate
```

5. Ensure that the iSeries user profile that you will use to log on to your iSeries system uses CCSID37:

- a. Log on to the iSeries system.

- b. Type the following command, where *user* is the user profile:

```
CHGUSRPRF USRPRF (user) CCSID(37)
```

- c. Make sure that the DDM server is started on the iSeries system type:

```
STRTCPSVR SERVER(*DDM)
```

6. Make sure that DB2 for Windows NT and DB2 for iSeries are connected:

```
db2 connect to rdb_name user user_name using password
```

---

## Authorizing user IDs for replication

If you have to access data in DB2 and non-DB2 relational servers, ensure that the following authorization requirements are met:

- “Authorization requirements for administration”
- “Authorization requirements for the Capture program” on page 19
- “Authorization requirements for Capture triggers on non-DB2 relational databases” on page 20
- “Authorization requirements for the Apply program” on page 21
- “Authorization requirements for the Replication Alert Monitor” on page 22

## Authorization requirements for administration

You use the Replication Center to administer replication (see Chapter 14, “Using the DB2 Replication Center” on page 243 for details). If your replication environment is only on the OS/400 operating system, you can use the OS/400 system commands to administer replication (see Chapter 18, “System commands for replication (OS/400)” on page 349 for details). To administer replication, you must have at least one user ID on all databases involved in the replication configuration and that user ID must have the

authority to set up replication. Your user ID does not need to be the same on all systems, although it would be easier for you if it was. Setting up replication involves creating objects (such as control tables and table spaces), binding plans (on UNIX, Windows, and z/OS), creating SQL packages (on OS/400), and running generated SQL to create tables, registrations, and subscription sets. You can use one authorized user ID on all servers in your replication environment, or you can use a different one on each server.

### **Requirements for UNIX, Windows, z/OS**

Ensure that the user IDs that you use to set up replication can perform the following tasks:

- Connect to all the servers (source server, Capture control server, Apply control server, Monitor control server, target server).
- Select from catalog tables on the source server, Capture control server, Monitor control server, and target server.
- Create tables (including replication control tables), table spaces, and views at the source server, Monitor control server, Capture control server, and Apply control server.
- If you use the DB2 Replication programs to create new target tables: Create tables and table spaces on the target server. (Not required if you use existing tables as targets).
- Bind plans or create packages on each DB2 database involved in replication, including the source server, target server, Monitor control server, and Apply control server.
- Create stored procedures using a shared library and call stored procedures (UNIX, Windows only).

For non-DB2 relational databases, the user ID must be able to do the following actions:

- Create tables.
- Create Capture triggers on source tables and control tables.
- Create procedures.
- Create nicknames on the DB2 federated database.
- Create sequences (for Oracle databases only).
- Select from catalog tables.

Most replication administrators have DBADM or SYSADM privileges. On DB2 for z/OS the replication administrator should be at least authorized to select from the catalog and should have all privileges necessary to create tables with the ASN schema and to create CD and target tables with the characteristics of the source tables, including index creation privileges.



## Requirements for OS/400

Ensure that the user IDs you use to set up replication can perform the following tasks:

- Connect to all the servers (source server, Capture control server, Apply control server, Monitor control server, target server).
- Select from catalog tables on the source server, Capture control server, Monitor control server, and target server.
- Create tables (including replication control tables) and views at the source server, Monitor control server, Capture control server, and Apply control server.
- If you use the DB2 Replication programs to create new target tables: Create tables on the target server. (Not required if you use existing tables as targets.)
- Bind plans or create packages on each DB2 database involved in replication, including the source server, target server, Monitor control server, and Apply control server.

Most replication administrators have DBADM or SYSADM privileges.

Use the Grant DPR Authority (**GRTDPRAUT**) command to authorize a user to register sources, subscribe to those sources, and create control tables. If you are replicating only between OS/400 systems, you should use the same user ID for all servers. See “GRTDPRAUT: Authorizing users (OS/400)” on page 402 for command syntax and parameter descriptions.

If the Grant DPR Authority (**GRTDPRAUT**) command is not installed on a machine, you must use the Grant Object Authority (**GRTOBJAUT**) command.

## Authorization requirements for the Capture program

The user ID that runs the Capture program must be able to access the DB2 system catalog, access and update all replication control tables on the Capture control server, and execute the Capture program packages. You can use the replication administrator user ID to run the Capture program, but this is not a requirement.

## Requirements for UNIX, Windows

Ensure that the user IDs that run the Capture program have the following authorities and privileges:

- DBADM or SYSADM authority.
- WRITE privilege on the capture path directory, because the Capture program creates diagnostic files in the **capture\_path** directory that you specify when you started the Capture program.

## Requirements for z/OS

The user ID used to run the Capture program *must* be registered with access to USS.

Also, ensure that the Capture load library is APF-authorized and that the user ID that runs the Capture program has the following authorities and privileges:

- SELECT, UPDATE, INSERT, and DELETE privileges for all replication tables on the Capture control server. (See “List of tables used at the Capture control server” on page 476 for a list of these tables.)
- SELECT privilege for the DB2 catalog (SYSIBM.SYSTABLES and SYSIBM.SYSCOLUMNS).
- TRACE privilege.
- MONITOR1 and MONITOR2 privilege.
- EXECUTE privilege for the Capture program packages.

Also, ensure that the user ID has WRITE access to the capture path directory (USS) or high-level qualifier (z/OS). To run the Capture program in the USS shell, the STEPLIB system variable must be set and it must include the capture load library. The HFS path, /usr/lpp/db2repl\_8\_1/bin, must be in your PATH.

## Requirements for OS/400

Use the Grant DPR Authority (**GRTDPRAUT**) command to authorize a user to run the Capture program on a local system. See “GRTDPRAUT: Authorizing users (OS/400)” on page 402 for command syntax and parameter descriptions. If you are replicating between only OS/400 systems, you should use the same user ID for all servers. If the **GRTDPRAUT** command is not installed on a machine, you must use the Grant Object Authority (**GRTOBJAUT**) command.

## Authorization requirements for Capture triggers on non-DB2 relational databases

If you are replicating from a non-DB2 RDBMS, Capture triggers are used to capture changes from the source. Remote user IDs (for example, from user applications) that change the remote source tables need authority to make inserts into the CCD table. In most cases, you do not need explicit authority to execute INSERT, UPDATE, or DELETE triggers because, after the triggers are defined on a table, the execution of the triggers is transparent to the application that is performing the INSERT, UPDATE, or DELETE. In the case of Informix databases, the remote user IDs that perform INSERT, UPDATE, and DELETE actions against the registered source table need EXECUTE PROCEDURE privilege.

## Authorization requirements for the Apply program

Ensure that the user ID that starts and operates the Apply program can run the program, read the password file, and has WRITE privileges to the apply path directory. Also, the user IDs in the password file are used to connect to other servers and those user IDs must have sufficient privileges to do the following tasks:

### Requirements for z/OS

Ensure that the user IDs that run the Apply program have the following authorities and privileges:

- The user ID used to run the Apply program *must* be registered with access to USS. The load library must be APF-authorized *only* if the Apply program is to be registered with ARM. To run the Apply program in the USS shell, the STEPLIB system variable must be set and it must include the apply load library. The HFS path, /usr/lpp/db2repl\_8\_1/bin, must be in your PATH.
- Execute the Apply program plan at the Capture control server, target server, and the Apply control server.

### Requirements for UNIX, Windows

Ensure that the user IDs that run the Apply program have the following authorities and privileges:

- Access the replication source tables (including associated CD and CCD tables)  
If your source tables are on a non-DB2 relational database management system: The user ID must have sufficient privileges in both the DB2 federated database *and* in the non-DB2 relational database to access the source tables through nicknames, which are defined on the federated database.
- Access and update the replication target tables.  
The Apply user ID must have update privileges for the target tables.  
If your target tables are on non-DB2 RDBMS: The user ID must have sufficient privileges in the DB2 federated database *and* in the non-DB2 relational database to access and update target tables through nicknames, which are defined on the federated database.
- Access and update all control tables that are generated by DB2 replication programs and built at the Capture control server and the Apply control server.
- Read any password file used by the Apply program.

### Requirements for non-DB2 relational database management systems

If your control tables are on non-DB2 relational database management systems, the user ID that is pushing changed data to a non-DB2 relational target or pulling data from it must have sufficient privileges in the DB2 federated database *and* in the non-DB2 relational database.

For non-DB2 relational targets, the user ID running the Apply program needs the privilege to WRITE to nicknames on the DB2 federated database and, through user mappings, the privilege to WRITE to the actual non-DB2 target.

For non-DB2 relational sources, the ID running the Apply program needs the following privileges:

- Privilege to READ from and WRITE to nicknames on the DB2 federated database and, through user mappings, the privilege to READ from and WRITE to the Capture control tables.
- Privilege to READ from nicknames on the DB2 federated database and, through user mappings, the privilege to READ from the actual CCD table on the non-DB2 server.
- Privilege to READ from nicknames on the DB2 federated database and, through user mappings, the privilege to READ from the actual source table on the non-DB2 server.

#### **Requirements for OS/400**

Use the Grant DPR Authority (**GRTDPRAUT**) command to authorize a user to run the Apply program on a local system. If you are replicating only between OS/400 systems, you should use the same user ID for all servers. If the **GRTDPRAUT** command is not installed on a machine, you must use the Grant Object Authority (**GRTOBJAUT**) command. See “GRTDPRAUT: Authorizing users (OS/400)” on page 402 for command syntax and parameter descriptions.

You can use different user IDs at each server in your replication environment.

### **Authorization requirements for the Replication Alert Monitor**

#### **Requirements for UNIX, Windows**

Ensure that the user ID that starts the Replication Alert Monitor is a valid logon ID on the Monitor control server, where the Monitor control tables reside, and on the servers that contain the control tables that you are monitoring. Also, ensure that the user ID that runs the Replication Alert Monitor has the following authority:

- SELECT, UPDATE, INSERT, and DELETE privileges for Monitor control tables on the Monitor control servers. (See “List of tables used at the Monitor control server” on page 481 for a list of these tables.)

- **SELECT** authority on the Capture and Apply control tables that reside on the servers that you want to monitor
- **BINDADD** authority (required only if you want to use the autobind feature for the monitor packages)

On UNIX and Windows, you need **WRITE** privilege on the **monitor\_path** directory where the Replication Alert Monitor stores diagnostic files, and you need read access to the password file used by the Replication Alert Monitor.

---

## Storing user IDs and passwords for replication (UNIX, Windows)

If your replication environment is not distributed across servers, you don't need to store user IDs and passwords. In most replication environments; however, data is distributed across servers. If you have such an environment, when you try to connect to a database, you must provide a valid user ID and password so that DB2 can verify your identity. You store the password information differently for the Replication Center and the other replication programs

You use the **asnpwd** command to create and maintain a password file so that the Apply program, the Replication Alert Monitor, and the Replication Analyzer can access data on remote servers. (The Capture program does not require a password file.) The information in the password file is encrypted to ensure confidentiality. See “asnpwd: Maintaining password files (UNIX and Windows)” on page 335 for command syntax and parameter descriptions.

For information about password requirements for the Replication Center see the Replication Center help and “Managing user IDs and passwords for the Replication Center” on page 247.

---

## Setting up the replication control tables

You can create control tables that are used for replication.

- “Creating control tables (UNIX, Windows)”
- “Creating control tables (z/OS)” on page 24
- “Creating control tables (OS/400)” on page 24
- “Creating control tables for non-DB2 relational sources” on page 24
- “Creating multiple sets of Capture control tables” on page 25

### Creating control tables (UNIX, Windows)

Use the Replication Center to create replication control tables for the Capture and Apply programs on UNIX and Windows. When you create the replication control tables, if you don't customize the way that the control tables are created, two table spaces are created, one for the UOW table and one for the

other control tables. If you do not want to use the default replication table spaces, you can specify existing table spaces, create new table spaces, or use the current DB2 default table space.

You can create separate profiles for UNIX and Windows operating systems to identify the defaults to be used when you create control tables for that type of system. After you set the profiles for these control tables, you do not have to set them for every set of control tables that you create; however, you can override the defaults when you create the control tables. You can also modify the profiles at any time, but the changes will affect only the control tables that you create *after* you modified the profiles.

### Creating control tables (z/OS)

Use the Replication Center to create replication control tables on z/OS. You can create a profile for z/OS platforms to identify the defaults to be used when you create control tables for that type of system. After you set the profiles for these control tables, you do not have to set them for every set of control tables that you create; however, you can override the defaults when you create the control tables. You can also modify the profile at any time, but the changes will affect only the control tables that you create *after* you modified the profile.

### Creating control tables (OS/400)

Replication control tables are created automatically when you install DB2 DataPropagator for iSeries. These tables are created in the DataPropagator default schema (called ASN), if they do not already exist.

You can create a new set of Capture control tables with a new Capture schema. You can create a maximum of 25 schemas. Use the Create DPR Tables (**CRTDPRTBL**) command, as described in “Creating multiple sets of Capture control tables” on page 25. You can also use the **CRTDPRTBL** command if your replication control tables are accidentally deleted or corrupted. For details about this command, see “**CRTDPRTBL**: Creating the replication control tables (OS/400)” on page 396.

**Important:** Use only the **CRTDPRTBL** command to create control tables on OS/400. The Replication Center does not support the creation of control tables for OS/400.

### Creating control tables for non-DB2 relational sources

If you want to replicate *from* a non-DB2 RDBMS, such as Informix, you must use the Replication Center to create control tables, just as you would if you are replicating from DB2. For these types of sources, the Replication Center creates the following Capture control tables in the non-DB2 relational database:

- Prune control table (IBMSNAP\_PRUNCNTL)

- Prune set table (IBMSNAP\_PRUNE\_SET)
- Register synchronization table (IBMSNAP\_REG\_SYNCH)
- Register table (IBMSNAP\_REGISTER)
- Sequencing table (IBMSNAP\_SEQTABLE), on Informix only
- Signal table (IBMSNAP\_SIGNAL)

Nicknames are created in a federated database for all but the sequencing table (IBMSNAP\_SEQTABLE). (The sequencing table is used only by the Informix triggers. The Apply program doesn't use it.) Triggers are created automatically on the signal table (IBMSNAP\_SIGNAL) and the register synchronization table (IBMSNAP\_REG\_SYNCH).

**Important:** Do not remove or modify the triggers that are created on the IBMSNAP\_SIGNAL and IBMSNAP\_REG\_SYNCH tables.

### Creating multiple sets of Capture control tables

If you want to use more than one Capture program on a server you must create more than one set of Capture control tables and ensure that each set of tables has a unique Capture schema. This schema identifies the Capture program that uses a set of tables. Multiple Capture schemas enable you to run multiple Capture programs concurrently.

You might want to run multiple Capture programs in the following situations:

- To optimize performance by treating low-latency tables differently from other tables. If you have low latency tables, you might want to replicate those tables with their own Capture program. That way, you can give them a different run-time priority. Also, you can set the Capture program parameters, such as pruning interval and monitor interval, to suit the low latency of these tables.
- To potentially provide higher Capture throughput. This may be a significant benefit in a source environment with multiple CPUs. The trade-off for the higher throughput is additional CPU overhead associated with multiple log readers.

If you want to replicate from multiple non-DB2 source databases within the same federated database, you must create multiple sets of Capture control tables, with each set having its own schema. Or, if you prefer, you can use separate federated databases, in which case the Capture control tables on each server can use the default ASN schema.

On z/OS systems, you can use multiple Capture schemas if you want to work with UNICODE and EBCDIC encoding schemes separately or if you want to run more than one instance of the Capture program on a subsystem. See “Creating control tables (z/OS)” on page 24 for information about creating control tables.

On OS/400 systems, use the Create DPR Tables (**CRTDPRTBL**) command to create the extra set of Capture control tables by using the **CAPCTLLIB** parameter to specify the schema name. For details about this command, see “CRTDPRTBL: Creating the replication control tables (OS/400)” on page 396.

---

## Setting up the replication programs

The following sections explain the steps involved in setting up the replication programs for the servers in your environment:

- “Setting up the replication programs (UNIX, Windows)”
- “Setting up the Capture and Apply programs (OS/400)” on page 30
- “Setting up the replication programs (z/OS)” on page 32

### Setting up the replication programs (UNIX, Windows)

Read the following instructions to set up the replication programs:

- “Setting environment variables for the replication programs (UNIX, Windows)”
- “Preparing the DB2 database to run the Capture program (UNIX, Windows)” on page 27
- “Optional: Binding the Capture program packages (UNIX, Windows)” on page 27
- “Optional: Binding the Apply program packages (UNIX, Windows)” on page 28
- “Optional: Binding the Replication Alert Monitor program packages (UNIX, Windows)” on page 29

### Setting environment variables for the replication programs (UNIX, Windows)

You must set environment variables before you start and stop the Capture program, the Apply program, or the Replication Alert Monitor program, and before you use the Replication Center or replication system commands.

#### Procedure:

To set the environment variables:

1. Set the environment variable for the DB2 instance name (DB2INSTANCE) as shown:

#### For Windows:

```
SET DB2INSTANCE=db2_instance_name
```

#### For UNIX:

```
export DB2INSTANCE=db2_instance_name
```



2. If you created the source database with a code page other than the default code page value, set the DB2CODEPAGE environment variable to that code page. See “Configuring National Language Support (NLS) for Replication” on page 14.<sup>4</sup>
3. Optional: Set environment variable DB2DBDFT to the source server.
4. **For UNIX:** Make sure the library path and executables path system variables specific to your system include the directory where the replication libraries and executables are installed.

## Preparing the DB2 database to run the Capture program (UNIX, Windows)

### Procedure:

To prepare the DB2 database to run the Capture program:

1. Connect to the Capture control server database by entering:

```
db2 connect to database
```

where *database* is the Capture control server database.

2. Prepare the Capture control server database for roll-forward recovery by issuing the **update database configuration** command (logretain recovery) and the **backup database** command. You might need to increase configuration values based on your installation requirements. For transactions with a large number of rows or very large rows it is recommended to increase the CAPPARMS memory limit parameter. The following database configuration values are adequate for many large workstation scenarios: APPLHEAPSZ 1000, LOGFILSZ 4000, LOGPRIMARY 8, LOGSECOND 40, DBHEAP 1000, LOGBUFSZ 16, MAXAPPLS 200.

### Optional: Binding the Capture program packages (UNIX, Windows)

The following steps are optional because the Capture program is bound automatically on UNIX and Windows during execution.

### Procedure:

To bind the Capture program packages:

1. Connect to the Capture control server database by entering:

```
db2 connect to database
```

where *database* is the Capture control server database.

---

4. Capture must be run in the same code page as the database for which it is capturing data. DB2 derives the Capture code page from the active environment where Capture is running. If DB2CODEPAGE is not set, DB2 derives the code page value from the operating system. The value derived from the operating system is correct for Capture if you used the default code page when creating the database.

2. Change to the directory where the Capture program bind files are located.

**Windows:**

drive:\sqllib\bnd

**UNIX:** *db2homedir*/sqllib/bnd

where *db2homedir* is the DB2 instance home directory.

3. Create and bind the Capture program package to the source server database by entering the following command:

```
db2 bind @capture.lst isolation ur blocking all
```

where *ur* specifies the list in uncommitted read format for greater performance.

These commands create packages, the names of which are in the file *capture.lst*.

**Optional: Binding the Apply program packages (UNIX, Windows)**

On UNIX and Windows the Apply program is bound automatically during execution. Therefore, the following steps are optional on those operating systems.

**Procedure:**

To bind the Apply program packages:

1. Change to the directory where the Apply program bind files are located.

**Windows:**

drive:\sqllib\bnd

**UNIX:** *db2homedir*/sqllib/bnd

where *db2homedir* is the DB2 instance home directory.

2. For each source server, target server, Capture control server, and Apply control server to which the Apply program connects, do the following steps:

- a. Connect to the database by entering:

```
db2 connect to database
```

where *database* is the source server, target server, Capture control server, or Apply control server. If the database is cataloged as a remote database, you might need to specify a user ID and password on the **db2 connect to** command. For example:

```
db2 connect to database user userid using password
```

- b. Create and bind the Apply program package to the database by entering the following commands:

```
db2 bind @applycs.lst isolation cs blocking all grant public
db2 bind @applyur.lst isolation ur blocking all grant public
```

where *cs* specifies the list in cursor stability format, and *ur* specifies the list in uncommitted read format.

These commands create packages, the names of which are in the files *applycs.lst* and *applyur.lst*.

### **Optional: Binding the Replication Alert Monitor program packages (UNIX, Windows)**

The following steps for binding the packages are optional. The Replication Alert Monitor packages are bound automatically during execution. If you want to specify options or check that all bind processes completed successfully, complete the following steps:

#### **Procedure:**

To bind the Replication Alert Monitor program packages:

1. Change to the directory where the Replication Alert Monitor program bind files are located.

#### **Windows:**

drive:\sqllib\bnd

**UNIX:** *db2homedir/sqllib/bnd*

where *db2homedir* is the DB2 instance home directory.

2. For each Monitor control server, do the following steps:

- a. Connect to the Monitor control server database by entering:

```
db2 connect to database
```

where *database* is the Monitor control server. If the database is cataloged as a remote database, you might need to specify a user ID and password on the **db2 connect to** command. For example:

```
db2 connect to database user userid using password
```

- b. Create and bind the Replication Alert Monitor program package to the database by entering the following commands:

```
db2 bind @asnmoncs.lst isolation cs blocking all grant public
db2 bind @asnmonur.lst isolation ur blocking all grant public
```

where *cs* specifies the list in cursor stability format, and *ur* specifies the list in uncommitted read format.

These commands create packages, the names of which are in the files `asnmoncs.lst` and `asnmonur.lst`.

3. For each server that you are monitoring and to which the Replication Alert Monitor program connects, do the following steps:

- a. Connect to the database by entering:

```
db2 connect to database
```

where *database* is the monitored server. If the database is cataloged as a remote database, you might need to specify a user ID and password on the **db2 connect to** command. For example:

```
db2 connect to database user userid using password
```

- b. Create and bind the Replication Alert Monitor program package to the database by entering the following command:

```
db2 bind @asnmonit.lst isolation ur blocking all grant public
```

where *ur* specifies the list in uncommitted read format.

These commands create packages, the names of which are in the file `asnmonit.lst`.

## Setting up the Capture and Apply programs (OS/400)

You must set up your environment if you want to use the Apply program with remote systems on other, non-OS/400 operating systems. The following sections explain the steps involved in setting up your replication environment:

- “Creating SQL packages to use with remote systems (OS/400)”
- “Granting privileges to the SQL packages” on page 31

### Creating SQL packages to use with remote systems (OS/400)

You need to create packages using the **CRTSQLPKG** command in the following cases:

- When using remote journaling. Run the **CRTSQLPKG** command on the system where the Capture program is running, point to the system where the source table is located.
- Before using the **ADDDPRSUB** or **ADDDPRSUBM** command to add a subscription set or subscription set member. Run the **CRTSQLPKG** command on the target server:
  - If the source table is on a different machine, point to the system where the source table is located.
  - If the Apply control server is on a different machine, point to the Apply control server.

The SQL packages allow replication programs to operate in a distributed replication environment, whether that environment is one in which you’re

replicating between OS/400 systems or between an OS/400 system and some other operating system (such as UNIX or Windows).

For information about using the **CRTSQLPKG** command, see *DB2 Universal Database for iSeries SQL Programming*.

The packages are created using the ASN qualifier. On OS/400 they are created in the ASN library. On other platforms they are created in the ASN schema.

**Creating SQL packages for the Apply program:** You must create SQL packages so that the Apply program can interact with all the remote servers to which it needs to connect. For example, run this command on the system where Apply is running to enable it to connect to a remote system:

```
CRTSQLPKG PGM(QDP4/QZSNAPV2) RDB(remote_system)
```

where *remote\_system* is the relational database entry name for the remote system to which the Apply program needs to connect.

**Creating SQL packages for the Replication Analyzer:** You must create SQL packages so that the Replication Analyzer can interact with the servers that you are analyzing, such as the Capture control server or the target server. Run this command on the system where the Replication Analyzer is running:

```
CRTSQLPKG PGM(QDP4/QZSNANZR) RDB(remote_system)
```

where *remote\_system* is the name of the system that you are analyzing.

**Creating SQL packages for the replication administration commands:** For replication between OS/400 systems, if you use a remote journal, you must use this command to create packages for the Capture program and for the replication administration commands. Run this command on the system where Capture is running:

```
CRTSQLPKG PGM(QDP4/QZSNSQLF) RDB(source_system) OBJTYPE(*SRVPGM)
```

where *source\_system* is the name of the system where the source table actually exists.

### Granting privileges to the SQL packages

After you create the packages, you must grant \*EXECUTE privileges to all users who will be subscribing to files registered on the source database. Log on to the OS/400 system where the source database resides and use one of the following methods:

- Use the Grant Object Authority (**GRTOBJAUT**) command:

```
GRTOBJAUT OBJ(ASN/package_name) OBJTYPE(*SQLPKG)  
USER(subscriber_name) AUT(*OBJOPR *EXECUTE)
```

- Use SQL to connect to the source database and run the GRANT SQL statement:  

```
CONNECT TO data_server RDB_name
GRANT EXECUTE ON PACKAGE ASN/package_name TO subscriber_name
```
- Use the **GRTDPRAUT** command, if it is installed on the local system. See “GRTDPRAUT: Authorizing users (OS/400)” on page 402 for command syntax and parameters.

## Setting up the replication programs (z/OS)

You must set up and customize the replication programs when you install IBM DB2 DataPropagator for z/OS. See the instructions in *Program Directory for IBM DB2 DataPropagator for z/OS*.

---

## Setting up journals (OS/400)

DB2 DataPropagator for iSeries uses the information that it receives from the journals about changes to the data to populate the CD and UOW tables for replication.

DB2 DataPropagator for iSeries runs under commitment control for most operations and therefore requires journaling on the control tables. (The QSQJRN journal is created when the **CRTDPRTBL** command creates a collection.)

Administrators must make sure the libraries containing the source table, CD table, and target table contain journals. They must also ensure that all the source tables are journaled correctly.

Before you register a table for replication on OS/400, the table must be journaled for both before-images and after-images.

The following sections describe the journal setup required for replication:

- “Creating journals for source tables (OS/400)”
- “Managing journals and journal receivers (OS/400)” on page 34

## Creating journals for source tables (OS/400)

To set up the source table journals, you must have the authority to create journals and journal receivers for the source tables to be defined. (Skip this section if your source tables are already journaled.)

**Important:** Use a different journal for the source tables than one of those created by DB2 DataPropagator for iSeries in the ASN (or other capture schema) library.

**Procedure:**

To create a source table journal:

1. Create a journal receiver in a library of your choice using the Create Journal Receiver (**CRTJRNRCV**) command. Place the journal receiver in a library that is saved regularly. Choose a journal receiver name that can be used to create a naming convention for future journal receivers, such as RCV0001. You can use the \*GEN option to continue the naming convention when you change journal receivers. This type of naming convention is also useful if you choose to let the system manage the changing of your journal receivers. The following example uses a library named JRNLIB for journal receivers.

```
CRTJRNRCV  JRNRCV(JRNLIB/RCV0001)
           THRESHOLD(100000)
           TEXT('DataPropagator Journal Receiver')
```

2. Create the journal by using the Create Journal (**CRTJRN**) command:

```
CRTJRN  JRN(JRNLIB/DJRN1)
        JRNRCV(JRNLIB/RCV0001)
        MNGRCV(*SYSTEM) DLTRCV(*YES)
        TEXT('DataPropagator Journal')
```

- Specify the name of the journal receiver that you created in step 1.
- Use the Manage receiver (MNGRCV) parameter to have the system change the journal receiver and attach a new one when the attached receiver becomes too large. If you choose this option, you do not need to use the **CRTJRN** command to detach receivers and create and attach new receivers manually.
- Use the default attribute MINENTDTA(\*NONE). Other values are not valid for this keyword.
- Specify DLTRCV(\*NO) only if you have overriding reasons to do so (for example, if you need to save these journal receivers for recovery reasons). If you specify DLTRCV(\*YES), these receivers might be deleted before you have a chance to save them.

You can use two values on the RCVSIZOPT parameter of the **CRTJRN** command (\*RMVINTENT and \*MINFIXLEN) to optimize your storage availability and system performance. See the *OS/400 Programming: Performance Tools Guide* for more information.

3. Start journaling the source table using the Start Journal Physical File (**STRJRNPf**) command, as in the following example:

```
STRJRNPf FILE(library/file)
        JRN(JRNLIB/DJRN1)
        OMTJRNE(*OPNCLO)
        IMAGES(*BOTH)
```

Specify the name of the journal that you created in step 2. The Capture program requires a value of \*BOTH for the IMAGES parameter.

4. Change the source table journaling setup:

- a. Use IMAGES(\*BOTH) to make sure that the source table is journaled for both before- and after-images.
- b. Make sure that the journal has the following attributes: MNGRCV(\*YES) and DLTRCV(\*YES).
- c. Make sure that the journal has MINENTDTA(\*NONE).

## Managing journals and journal receivers (OS/400)

The Capture program uses the Receive Journal Entry (**RCVJRNE**) command to receive journals. The following sections describe how to manage journals and journal receivers in your replication environment:

- “Specifying system management of journal receivers (OS/400)”
- “Changing definitions of work management objects (OS/400)”
- “Specifying user management of journal receivers” on page 35
- “Using the delete journal receiver exit routine” on page 35

### Specifying system management of journal receivers (OS/400)

**Recommendation:** Let the OS/400 system manage the changing of journal receivers. This is called *system change journal management*. Specify MNGRCV(\*SYSTEM) when you create the journal, or change the journal to that value. If you use system change journal management support, you must create a journal receiver that specifies the threshold at which you want the system to change journal receivers. The threshold must be at least 5 000 KB, and should be based on the number of transactions on your system. The system automatically detaches the receiver when it reaches the threshold size and creates and attaches a new journal receiver if it can.

### Changing definitions of work management objects (OS/400)

When you install DB2 DataPropagator for iSeries, the installation program creates an SQL journal, an SQL journal receiver for this library, and work management objects. Table 1 lists the work management objects that are created.

Table 1. Work management objects

Description	Object type	Name
Subsystem description	*SBSD	QDP4/QZSNDPR
Job queue	*JOBQ	QDP4/QZSNDPR
Job description	*JOB D	QDP4/QZSNDPR

You can alter the default definitions for the three types of work management objects or provide your own definitions. If you create your own subsystem description, you must name the subsystem QZSNDPR and create it in a library other than QDP4. See *iSeries Work Management, SC41-5306* for more information about changing these definitions.



### Specifying user management of journal receivers

If you specify MNGRCV(\*USER) when you create the journal (meaning you want to manage changing your own journal receivers), a message is sent to the journal's message queue when the journal receiver reaches a storage threshold, if one was specified for the receiver.

Use the **CHGJRN** command to detach the old journal receiver and attach a new one. This command prevents Entry not journaled error conditions and limits the amount of storage space that the journal uses. To avoid affecting performance, do this at a time when the system is not at maximum use.

You can switch journal receiver management back to the system by specifying CHGJRN MNGRCV(\*SYSTEM).

You should regularly detach the current journal receiver and attach a new one for two reasons:

- Analyzing journal entries is easier if each journal receiver contains the entries for a specific, manageable time period.
- Large journal receivers can affect system performance and take up valuable space on auxiliary storage.

The default message queue for a journal is QSYSOPR. If you have a large volume of messages in the QSYSOPR message queue, you might want to associate a different message queue, such as DPRUSRMSG, with the journal. You can use a message handling program to monitor the DPRUSRMSG message queue. For an explanation of messages that can be sent to the journal message queue, see *OS/400 Backup and Recovery*.

### Using the delete journal receiver exit routine

When you install DB2 DataPropagator for iSeries, a *delete journal receiver* exit routine (**DLTJRNRCV**) is registered automatically. This exit routine is called any time a journal receiver is deleted, whether or not it is used for journaling the source tables. This exit routine determines whether or not a journal receiver can be deleted.

To take advantage of the delete journal receiver exit routine and leave journal management to the system, specify DLTRCV(\*YES) and MNGRCV(\*SYSTEM) on the **CHGJRN** or **CRTJRN** command.

**Important:** If you remove the registration for the delete journal receiver exit routine, you must change all the journals used for source tables to have the DLTRCV(\*NO) attribute.

If the journal that is associated with the receiver is not associated with any of the source tables, this exit routine *approves* the deletion of the receiver.

If the journal receiver is used by one or more source tables, this exit routine makes sure that the receiver being deleted does not contain entries that have not been processed by the Capture program. The exit routine *disapproves* the deletion of the receiver if the Capture program still needs to process entries on that receiver.

If you must delete a journal receiver and the delete journal receiver exit routine does not approve the deletion, specify DLTJRNRCV DLTOPT(\*IGNEXITPGM) to override the exit routine.

**Removing the delete journal receiver exit routine:** If you want to handle the deletion of journal receivers manually, you can remove the delete journal receiver exit routine by using the following command:

```
RMVEXITPGM EXITPNT (QIBM_QJO_DLT_JRNRCV)
              FORMAT(DRCV0100)
              PGMNBR(value)
```

#### **Procedure:**

To determine the PGMNBR value for the **RMVEXITPGM** command:

1. Issue the **WRKREGINF** command.
2. On the Work with Registration Information window, find the entry for exit point QIBM\_QJO\_DLT\_JRNRCV. Enter 8 in the **Opt** field.
3. On the Work with Exit Programs window, find the entry for Exit Program QZSNDREP in library QDP4. The number that you need is under the Exit Program Number heading.

**Registering the delete journal receiver exit routine:** Use the **ADDEXITPGM** command if you removed the exit point and want to put it back. You must register the exit routine with this command:

```
ADDEXITPGM EXITPNT(QIBM_QJO_DLT_JRNRCV)
              FORMAT(DRCV0100)
              PGM(QDP4/QZSNDREP)
              PGMNBR(*LOW)
              CRTEXITPNT(*NO)
              PGMDTA(65535 10 QSYS)
```

---

## Chapter 3. Registering tables and views as replication sources

With DB2 replication, you identify the tables and views that you want to use as replication sources by registering them. When you register a particular table or view for replication, you create a source of available data that you can later use with different targets for various purposes. The administration tasks described in this chapter help you set up the control information that defines how data is captured from each source based on your replication goals.

When you register a source, you identify the table or view that you want to use as a replication source, which table columns you want to make available for replication, and the properties for how DB2 replication captures data and changes from the source.

For DB2 replication, you can register the following objects as sources:

- A DB2 table
- A non-DB2 relational table through a nickname
- A subset of the data in a table (DB2 or non-DB2 relational)
- A view over a single table (DB2)
- A view that represents an inner join of two or more tables (DB2)

This chapter contains the following sections:

- “Registering DB2 tables as sources”
- “Registering non-DB2 relational tables as sources” on page 39
- “Registration options for source tables” on page 41
- “How views behave as replication sources” on page 58
- “Registering views of tables as sources” on page 60
- “Maintaining CCD tables as sources (IMS)” on page 61

---

### Registering DB2 tables as sources

This section describes how to register DB2 tables as replication sources. DB2 replication supports the following types of DB2 tables as sources:

#### For UNIX and Windows

- DB2 tables that your application maintains
- Catalog tables (for full-refresh-only replication)
- Automatic summary tables

- External CCD tables

#### **For z/OS**

- DB2 tables that your application maintains
- Catalog tables
- External CCD tables

#### **For OS/400**

- DB2 tables that your application maintains (locally or remotely journaled)
- External CCD tables

For all DB2 sources except for OS/400, the source table DDL requires the DATA CAPTURE CHANGES option. Do not remove this option from your source.

When you register a table as a source, a CD (change-data) table is created for you. The Capture program that is associated with the registered table reads the log for the source and stores inflight changes that occur for registered columns in memory until the transaction commits or rollback. For a rollback, the changes are deleted from memory. For a commit, the changes are inserted into the CD table as soon as the Capture program reads the commit log record. Those changes are left in memory until the Capture program commits the changes after each Capture cycle. The Capture program does not start capturing data for a DB2 source table until a CAPSTART signal has been issued, either by you or the Apply program.

**Note for non-relational source tables:** You can register DB2 tables that contain data from non-relational database management systems, such as IMS. To do this, you need an application, such as IMS DataPropagator or Data Refresher, to populate a CCD table with the data from the non-relational database. The application captures changes to the non-relational segments in the IMS database and populates a CCD table. The CCD table must be complete, but it can be either condensed or non-condensed. Like other CCD sources, there is no Capture program that is associated with a CCD source table because the table already stores changed data from the non-relational source table. IMS DataPropagator and Data Refresher products maintain the values in the register (IBMSNAP\_REGISTER) table so that the Apply program can read from this source table correctly. If you do not use one of these products to maintain these types of CCD tables and you maintain the tables yourself, see “Maintaining CCD tables as sources (IMS)” on page 61.

#### **Prerequisites:**

Capture control tables must already exist on the Capture control server that will process the table that you want to register as a source. If you need to create Capture control tables, see “Setting up the replication control tables” on page 23.

**Restrictions (OS/400):**

- Because SQL statements are limited to a length of 32,000 characters, you can register only approximately 2000 columns per table; the exact number of columns depends on the length of the column names.
- For a single Capture schema, do not register more than 300 source tables that use the same journal.

**Procedure:**

Use one of the following methods to register a DB2 table:

**Replication Center**

Use the Register Tables window. See the Replication Center help for details.

**Tip:** To save time when registering, you can set up a source object profile ahead of time for the Capture control server. When you register a table, the Replication Center then uses the defaults that you defined in the source object profile instead of the Replication Center defaults. This can save you time when registering because you can overwrite the defaults once instead of having to select each table one at a time and change the default settings manually.

**System commands for replication (OS/400)**

Use the **ADDDPRREG** system command. See “ADDDPRREG: Adding a DPR registration (OS/400)” on page 349 for the syntax of this command and a description of its parameters.

When you register a DB2 table, you identify which table you want to register by specifying the source server, source table name, and the Capture schema. You can register the same table multiple times using different Capture schemas. You can use the default settings for registration, or you can modify the registration options to meet your replication needs. See “Registration options for source tables” on page 41 for a complete list of registration options, their defaults, and an explanation of when you might want to use or change the defaults.

---

## Registering non-DB2 relational tables as sources

This section describes how to register non-DB2 relational tables as replication sources. DB2 replication accesses non-DB2 relational tables by using nicknames.

When you register a non-DB2 relational table as a source, a CCD (consistent-change data) table is created for you. When a change for a registered non-DB2 relational table occurs, the Capture triggers simulate the Capture program and insert the change in the CCD table. The Capture triggers start capturing changes for a non-DB2 relational source table at the time you register the source.

By default, the CCD owner is derived from the schema name of the source table. If you modify the CCD owner so that it does not match the schema name, make sure that the source table owner is authorized to write to the CCD table. If the source table owner cannot update the CCD table, triggers on the source table will not be able to write changes to the CCD table.

**Prerequisite:**

Capture control tables must already exist on the Capture control server that will process this source. If you need to create Capture control tables, see “Creating control tables for non-DB2 relational sources” on page 24.

**Restrictions:**

- If you are using a single federated DB2 database to access multiple non-DB2 relational source servers, you must use a different Capture schema for *each* non-DB2 relational source server on that single federated database; no two can be the same. You can register a non-DB2 relational table under only one Capture schema.
- You cannot register columns in non-DB2 relational tables that have data types of LOB or DATALINK. If you register a table that includes these data types, you must register a column subset. See “Registering a subset of columns (vertical subsetting)” on page 42 for details on how to register only a subset of columns.

**Procedure:**

To register a non-DB2 relational table:

**Replication Center**

Use the Register Nicknames window. See the Replication Center help for details.

**Tip:** To save time when registering, you can set up a source object profile ahead of time for the Capture control server. When you register a table, the Replication Center then uses the defaults that you defined in the source object profile for CCD tables and nicknames for CCD tables instead of the Replication Center defaults. This can save

you time when registering because you can overwrite the defaults once instead of having to select each table one at a time and change the default settings manually.

When you register a non-DB2 relational table, you identify which table you want to register by specifying the nickname of the source table. You can use the default settings for registration, or you can modify the registration options to meet your replication needs. See “Registration options for source tables” for a complete list of registration options, their defaults, and an explanation of when you might want to use or change the defaults.

---

## Registration options for source tables

This section discusses the various options that you have when registering a table as a replication source. These options are part of the larger task of registering a table. For information on how to register:

- A DB2 table, see “Registering DB2 tables as sources” on page 37.
- A non-DB2 relational table, see “Registering non-DB2 relational tables as sources” on page 39.

When you create views over tables and register the views as sources, the views’ registration options are determined by the registration definition of the underlying tables. For details on which characteristics views inherit from the underlying tables and how views behave according to the underlying registration definition, see “How views behave as replication sources” on page 58.

After you choose which table that you want to register, you can identify which columns you want to make available for replication, and you can define properties that determine how registered data from this source will be handled and stored. You can also specify other registration options, such as how you want the Capture program to store source data in the CD table (or how you want the Capture triggers to store data in the CCD table). This section discusses the following options that you can specify when registering tables as sources:

- “Registering a subset of columns (vertical subsetting)” on page 42
- “Full-refresh copying and change-capture replication” on page 42
- “After-image columns and before-image columns” on page 44
- “Before-image prefix” on page 47
- “Stop the Capture program on error” on page 48
- “How the Capture program stores updates” on page 48
- “Preventing the recapture of changes (update-anywhere replication)” on page 49

- “Setting conflict detection (update-anywhere replication)” on page 54
- “Using relative record numbers (RRN) instead of primary keys (OS/400)” on page 57

## Registering a subset of columns (vertical subsetting)

**Default:** All columns are registered for replication

When you define a source table for replication, you don't need to register all the columns in the table for replication; you can register a subset of the columns in your source table. This vertical subset can be useful if you do not want to make all of the columns in the table available for targets to subscribe to. You might also want to select this option if the target tables for this source do not support all data types that are defined for the source table.

To register a subset of the columns, select *only* those columns that you want to make *available* for replication to a target table. The columns that you do not select will not be available for replication to any target table. Because CD (and CCD) tables must contain sufficient key data for some types of target tables (such as point-in-time), make sure that your subset contains the columns that will act as the key columns (primary key or unique index) for the target.

**Tip:** Register a subset of the columns in the source table only if you are sure that you will never want to replicate the unregistered columns. If you register a subset of the columns at the source and later want to replicate columns that you didn't register, you must then alter your registrations to add unregistered columns. (For non-DB2 relational sources, you must redefine your registrations altogether to add new columns to a registration.) If you plan to have an internal CCD associated with this source, it can be even more difficult to add columns later because registering new columns adds them to the CD table but not the internal CCD. To avoid these problems, you might want to register *all* columns at the source and use the Apply program instead to subset which columns are replicated to the targets. For details on how to subset at the target instead of the source, see “Source columns that you want applied to the target” on page 91.

## Full-refresh copying and change-capture replication

**Default:** Change-capture replication

You can choose whether you want all the data in the source table to be replicated to the targets during each replication cycle (full-refresh-only replication), or whether you want only the changes that occurred since the last time that the targets were updated (change-capture replication).

**Restriction:** You can capture changes on a multi node DB2 Enterprise Server Edition (ESE) configuration *only* if the source table is nonpartitioned and it resides on the catalog node.



### **Full-refresh only replication**

When targets subscribe to a source that is registered for full-refresh only replication, the Apply program deletes all data from the target table, copies the data that is in the registered columns at the source, and populates the targets with the source data during each replication cycle. The Capture program is not involved, and there is no CD table; the Apply program reads data directly from the source table.

**Tip for smaller tables:** You might want to choose full-refresh only replication if you have a very small source table that does not take much time or resources to copy.

**Tip for larger tables:** If you have larger tables and want to use full-refresh only replication, you might want to use the ASNLOAD exit routine to load your tables faster. See “Refreshing target tables using the ASNLOAD exit routine” on page 154 for details.

**Restriction:** If you plan to have a condensed target table that subscribes to this source and you cannot come up with a unique index for that target table, you must register the source for full-refresh only replication.

### **Change-capture replication**

**Default:** Changes to all rows are captured

During change-capture replication, only changed data is replicated to the target table. Depending on the type of target table you choose for this source, you must perform an initial load of the table. In most cases, the Apply program performs an initial full refresh, and then continues with change-capture replication.

If you choose not to allow full refresh for target tables, you must manually reload the table if the source and target tables need to be resynchronized. After the target is loaded with the initial source data, the Capture program captures changes that occur at the source and stores them in the CD table. In change-capture replication for non-DB2 relational sources, the Capture triggers capture changes at the source and store them in the CCD table. The Apply program reads the changes from the CD or CCD table and applies the changes to the targets that subscribe to the registered source.

When you define a DB2 source table for change-capture replication, you might not want to store all changes that occur at the source in the CD table. You can register a row (horizontal) subset that filters the changes so that fewer are captured in the CD table than actually occur at the source. You can select from the following two row-capture rules to determine which changed rows from the source table the Capture program records in the CD table:

- Changes to all rows are captured.

- Changes are captured only if the change occurred in a registered column. (DB2 only)

By default, changes are captured whenever a row is updated for any column (registered or unregistered) at the source table. If you register only a subset of the columns, the Capture program records the row values for the registered columns in the CD table every time a change occurs to the source table, even if the columns that changed are different from the registered columns. Use this default option if you want to keep a history of all changes to the source table. This is the *only* option available for non-DB2 relational sources, the Capture triggers capture all changed rows at the source, even if the change occurs in an unregistered column.

**Example:** Assume that you have 100 columns in your table and you register 50 of those columns for replication. By default, any time a change is made to any of the 100 columns in your table, the Capture program will write a row to the CD table (or the Capture triggers will write a row to the CCD table).

If you have a DB2 source, you might want the Capture program to capture changes for registered columns only. In this case, the Capture program writes a row to the CD table *only* when changes occur to registered columns.

**Suggestion:** Choose to capture changes for all rows if you need information for auditing purposes, or if changes in the table almost always occur in registered columns only. Choose to capture changes for only registered columns if changes frequently occur that only affect unregistered columns. Use this option if you don't want to keep a history of all changes to the source table.

## After-image columns and before-image columns

**Default:** After-image columns only

When you are registering a source for change-capture replication, you can choose whether you want the Capture program to capture only the after-image value (the value in the column after a change was made) or both the after-image value and the before-image value (the value that was in the column before the change was made). For UNIX, Windows, and z/OS, you can select whether to capture before-image values for each column in the table. For OS/400, you can select whether to capture before images for all or none of the columns in the table; you cannot select this option for each individual column. The sections below discuss when you should choose each option.

Several non-DB2 relational source tables require you to include only after-image values in the CCD table:

- A Sybase or Microsoft SQL Server table can contain only one column of type `TIMESTAMP`. When the data source is Sybase or Microsoft SQL Server and the source table has a column of type `TIMESTAMP`, select after images only for this column when you define it as part of the replication source.
- An Oracle table can contain only one column of type `LONG`. When the data source is Oracle and the source table has a column of type `LONG`, select after images only for this column when you define it as part of the replication source.

Columns with certain data types do not allow you to include before-image values in the CD table:

- Columns with LOB data types
- Columns with DATALINK data types

### **Capturing after-image values only**

For each column that you register for change-capture replication, you can choose for the Capture program or triggers to record only the after-image value for each change. When you select to capture after-image values only, the CD (or CCD) table contains one column for each changed value, which stores the value of the source column after the change occurred.

You don't need before images if you plan to use only base aggregate and change aggregate target-table types for this source. Before-image columns do not make sense if you plan to use your target table for computed values because there is no before image for computed columns. All other target-table types can make use of before-image columns. See "A computed summary of data or changes at the source" on page 81 for more information on aggregate target tables.

### **Capturing before-image and after-image values**

For each column that you register for change-capture replication, you can choose for the Capture program or triggers to record both the before-image and after-image value for each change. When you select to capture before-image and after-image values, the CD (or CCD) table contains two columns for each changed value: one for the value in source column before the change occurred, and one for the value after the change occurred.

When you choose to store both the before and after images in the CD (or CCD) table, the before-image columns and after-image columns have different values for different actions performed on the source tables:

<b>Action</b>	<b>Column value</b>
<b>Insert</b>	The before-image column contains a NULL value. The after-image column contains the inserted value.
<b>Update</b>	The before-image column contains the column value before

the change occurred. The after-image value contains the column value after the change occurred.

When you choose to have updates captured as delete and insert pairs, the delete row contains the before image from the update in both the before-image and after-image columns of the row, and the insert row contains NULL values in the before-image column and the after image in the after-image column. See “How the Capture program stores updates” on page 48 for more information on this option.

**Delete** The before-image and after-image columns contain the column value before the change occurred.

**Important for UNIX, Windows, and OS/400:** For columns that have before-images defined, DB2 replication limits column names to 29 characters because the entire column name can have only 30 characters. If the column name is longer, DB2 replication truncates the additional characters from the right by default, unless you have set your profile to truncate from the left. Because DB2 replication adds a before-image column identifier (usually X) to target columns and each column name must be unique, you cannot use column names that are longer than 29 characters. For tables that you do not plan to replicate, you can use longer column names, but consider using 29-character names in case you might want to replicate these columns in the future.

**Important for z/OS:** For tables in DB2 for z/OS, you can use 18-character column names, but DB2 DataPropagator will replace the 18th character with the before-image column identifier in target tables, so you must ensure that the first 17 characters of the name are unique.

The following sections describe cases in which you might want to capture before-image values:

- “For keeping a history of your source data”
- “For update-anywhere configurations with conflict detection”
- “When the key columns at the target are subject to update” on page 47

**For keeping a history of your source data:** If you want to keep data for auditing purposes, you might want to select both before and after images so that you have a record of how the data has changed over a period of time. A set of before-image and after-image copies is useful in some industries that require auditing or application rollback capability.

**For update-anywhere configurations with conflict detection:** In update-anywhere configurations where conflicts are possible between replica tables (where conflict detection is set to anything other than None), you must

register both after-image and before-image columns for the CD table of the replicas so that changes can be rolled back if a conflict occurs.

**When the key columns at the target are subject to update:** When registering a source, consider the potential target tables that you might define using this table as the source. Typically target tables are condensed and require a column or set of columns that make each row in that target table unique. These unique columns make up what is called the target key. If any of these target key columns might be updated at the source, DB2 replication requires special handling to ensure that the correct rows at the target table are updated. To ensure that DB2 replication updates the correct rows in the target table with the new key value, you can select to capture both after-images and before-images for the columns that will make up the target key. The Apply program needs the before-image values for these registered columns when it applies the changes of non-key source columns to target key columns in the target table. When applying the changes, the Apply program searches in the target table for the row by looking for the target key values that match the before-image value in the source's CD (or CCD) table, and then it updates that target row with the after-image value in the source's CD (or CCD) table.

Although you register these before-image values when you register the source table or view, DB2 replication does not know that your application will make updates to the target key. Later when you define which targets subscribe to this source (by creating subscription sets), you can specify for the Apply program to perform special updates when applying changes from non-key columns at the source to key columns at the target. See "How the Apply program updates the target key columns with the target-key change option" on page 94 for more information.

## **Before-image prefix**

**Default (Replication Center):** X

**Default (OS/400 system commands):** @

If you choose to capture both after-image and before-image columns in the CD (or CCD) table, the after-image column name is the name of the column at the source table, and the before-image column name is the name of the column at the source table with a one-character prefix added to the beginning. You can change the default one-character prefix for the before-image column names. The combination of the before-image prefix and the CD (or CCD) column name must be unambiguous, meaning that a prefixed column name cannot be the same as a current or potential column name in the CD (or CCD) table.

**Example:** If you use X as your before-image prefix and you register a source column named COL, you cannot register a column named XCOL because it is

unclear whether XCOL is an actual column name of another source column, or the name of a before-image column with a column name of COL and a before-image prefix of X.

If you are not replicating any before-image columns for a table, you can choose not to have a before-image prefix and set this property to null.

## **Stop the Capture program on error**

**Default:** The Capture program stops when it encounters certain errors

When the Capture program detects certain problems when processing registrations, it can either terminate or continue to run. You can choose one of the following options to determine how the Capture program reacts when it encounters certain errors while processing a registered source:

### **Stop Capture on error**

The Capture program writes an error message in the Capture trace (IBMSNAP\_CAPTRACE) table and terminates.

### **Do not stop Capture on error**

The Capture program continues to run when certain errors occur. If it encounters errors during the first time trying to process the source, it does not activate the registration. If the registered source was already active, it stops processing the registration.

This option determines whether the Capture program stays up when the following non-fatal errors occur:

- The registration is not defined correctly.
- The Capture program did not find the CD table when it tried to insert rows of changed data.
- The DATA CAPTURE CHANGES option on the (non-OS/400) source table was detected as being turned OFF when the Capture program was started or reinitialized.

For fatal errors, the Capture program does not continue running.

## **How the Capture program stores updates**

**Default:** Updates are stored in a single row in the CD table

You can choose how source updates are stored in the CD (or CCD) table. When the Capture triggers or Capture program captures an update to the source table, it can either save the updated value in a single row in the CD table, or it can store the delete in one row and the insert in another, using two rows in the CD (or CCD) table. By default, the update is stored in a single row. You can use this default to reduce storage and increase performance because only one row is stored in the CD (or CCD) table and is read by the Apply program for each change. However, there are some scenarios where

you should instruct the Capture program or triggers to capture updates to the source table as DELETE and INSERT pairs.

You *must* capture updates as DELETE and INSERT statements when your source applications update one or more columns referenced by a predicate on the subscription-set member. Suppose that you plan to define a target that subscribes only to source data with a predicate based on a specific column value (for example, WHERE DEPT = 'J35'). When you change that column (for example, to DEPT='FFK'), the captured change will not be selected for replication to the target because it does not meet the predicate criteria. That is, your new FFK department will not be replicated because your subscription-set member is based on department J35. Converting the updates to a DELETE and INSERT pair ensures that the target-table row is deleted.

Each captured update is converted to two rows in the CD (or CCD) table for all columns. You might need to adjust the space allocation for the CD (or CCD) table to accommodate this increase in captured data.

**Important for DATALINK values:** For DATALINK columns defined as ON UNLINK DELETE, the unlink is ignored because a DELETE and INSERT pair is handled within the same transaction. The external file is not deleted, but is updated.

## **Preventing the recapture of changes (update-anywhere replication)**

**Default for new source table:** Recapture changes

**Default for new replica table:** Do not recapture changes

**Restriction:** Tables from non-DB2 relational databases cannot participate in update-anywhere; therefore, this option is for only DB2 sources.

In update-anywhere replication, changes can originate at the master table or at the associated replica tables. When you register a table that you plan to use in update-anywhere replication, DB2 replication assumes that it will be the master table in your configuration. You can use the recapture option to control whether changes that originate from one site and are replicated to a second site are recaptured at that second site and therefore available to be replicated to additional sites. During registration, you set this option for the master table. Later, when you map the master source table with its replica targets, you can set whether changes at the replica are recaptured and forwarded to other tables. (For details about mapping masters to replicas, see “Defining read-write targets (update-anywhere)” on page 87.)

When you are registering the source table that will act as the master in your update-anywhere configuration, you can choose from the following two options:

**Recapture changes at master**

Updates to the master that originated at a replica are recaptured at the master and forwarded to other replicas.

**Do not recapture changes at master**

Updates to the master that originated at a replica are not recaptured at the master and forwarded to other replicas.

When you are registering the replica table in your update-anywhere configuration, you can choose from the following two options:

**Recapture changes at replica**

Updates to the replica that originated at the master are recaptured at the replica and forwarded to other replicas that subscribe to this replica.

**Do not recapture changes at replica**

Updates to the replica that originated at the master are not recaptured at the replica and forwarded to other replicas that subscribe to this replica.

Preventing changes from being recaptured can increase performance and reduce storage costs because the Capture program isn't capturing the same changes again for each replica.

The following sections discuss how to decide whether to recapture changes based on your update-anywhere configuration:

- “For masters with only one replica”
- “For multiple replicas that are mutually exclusive partitions of the master” on page 51
- “For masters that replicate changes to multiple replicas” on page 51
- “For replicas that replicate changes to other replicas (multi-tier)” on page 52

**For masters with only one replica**

**Master:** Do not recapture changes at master

**Single replica:** Do not recapture changes at replica

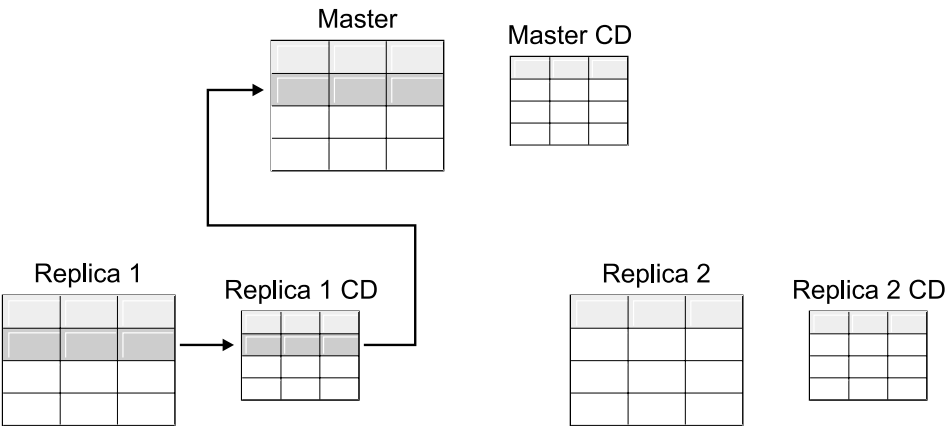
If you plan to have only one replica in your update-anywhere configuration, you might want to prevent changes from being recaptured at both the master and replica tables. This setting is optimal if the master table is not a source for other replica tables and the replica is not a source for other replicas (in a multi-tier configuration). If there are only these two tables involved, then a change that originates at the replica does not need to be recaptured at the master, and any change that originates at the master does not need to be recaptured at the single replica.



**For multiple replicas that are mutually exclusive partitions of the master**  
**Master:** Do not recapture changes at master

**Replicas:** Do not recapture changes at replicas

If you plan to have several replicas that are partitions of the master table, you might want to prevent changes from being recaptured at both the master and each replica. This setting is optimal if none of the replicas is a source for other replica tables. When replicas are partitions of the master, no two replicas ever subscribe to the same data at the master. Therefore, any change that originates at any replica does not need to be recaptured at the master and forwarded on to the other replicas because only the replica where the change occurred subscribes to that source data.

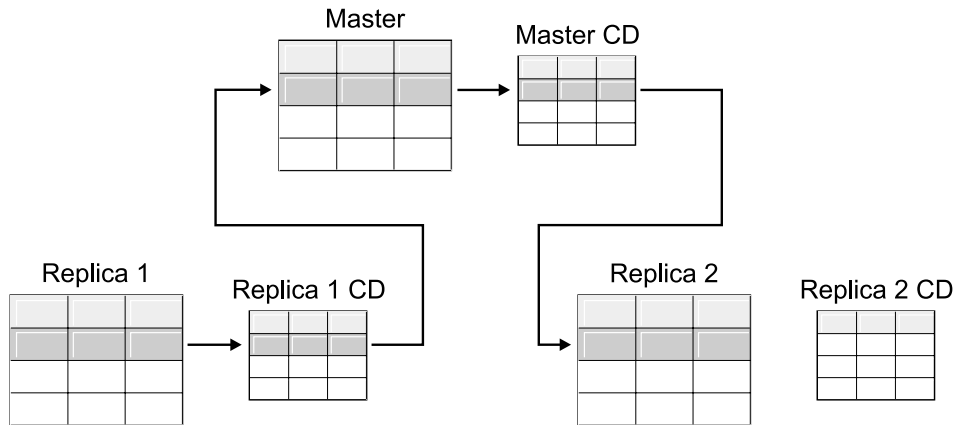


*Figure 1. Recapture option for replicas that are mutually exclusive partitions of the master. When you have multiple replicas that do not subscribe to the same data in the master, you do not need to use the recapture option for any of the tables.*

**For masters that replicate changes to multiple replicas**  
**Master:** Recapture changes at master

**Replicas:** Do not recapture changes at replicas

If you plan to have several replicas that subscribe to the same data in the master table, you might want the Capture program to recapture changes at the master. Changes that originate at a replica are then recaptured at the master and replicated down to other replicas that subscribe to the updated master data.



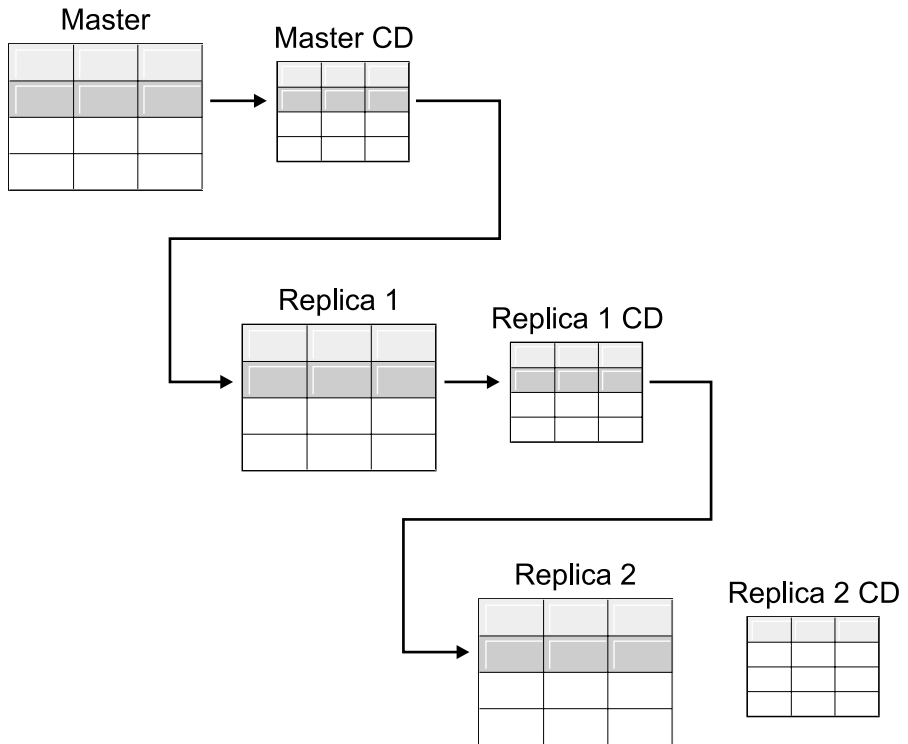
*Figure 2. Recapture option for masters that replicate changes to multiple replicas. When you have multiple replicas that subscribe to the same data in the master, you can use the recapture option at the master so that changes that occur at one replica are recaptured at the master and forwarded to the other replica tables.*

### **For replicas that replicate changes to other replicas (multi-tier)**

**Master:** Do not recapture changes at master

**Replicas:** Recapture changes at replicas

You can have a multi-tier configuration in which the master (tier 1) acts as a source to a replica (tier 2), and then that replica also acts as a source to another replica (tier 3). If you plan to have this type of configuration, you might want the Capture program to recapture changes at the middle replica (tier 2) so that changes that originated at the master are forwarded to the next replica (tier 3).



*Figure 3. Recapture option at tier 2 allows changes at tier 1 to be replicated down to tier 3. When you have a replica table that acts as a middle tier in a multi-tier configuration, you can use the recapture option at the replica so that changes that occur at the master are recaptured at the replica in the middle tier and forwarded to the replica in the subsequent tier.*

Also, when you have recapture set for the middle replica (tier 2), changes that originate at the final replica (tier 3) are recaptured at the middle replica (tier 2) and forwarded to the master (tier 1).

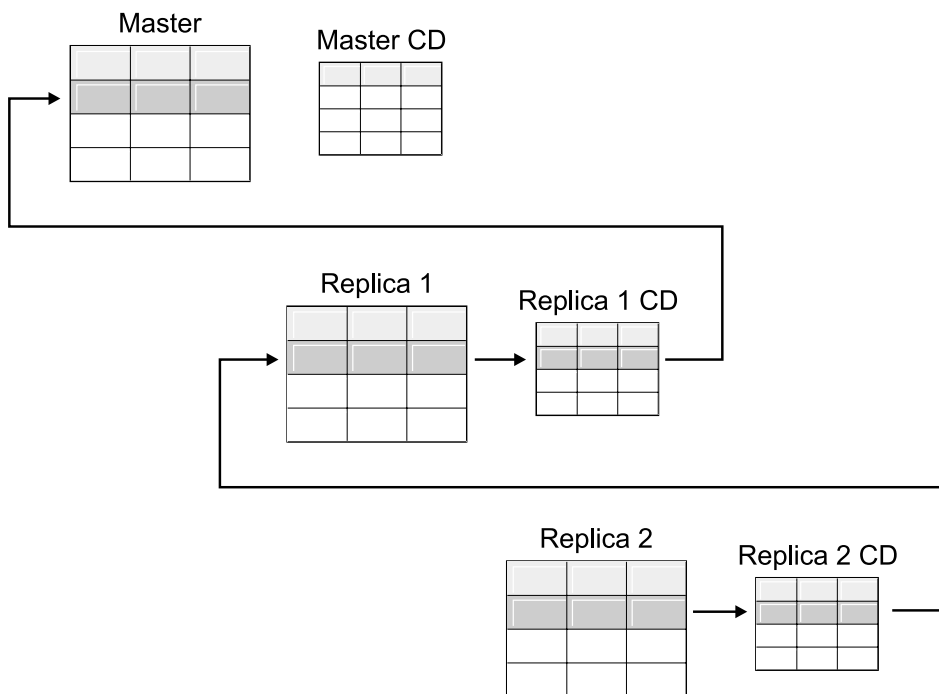


Figure 4. Recapture option at tier 2 allows changes at tier 3 to be replicated up to tier 1. When you have a replica table that acts as a middle tier in a multi-tier configuration, you can use the recapture option at the replica so that changes that occur at the replica in the subsequent tier are recaptured at the replica in the middle tier and forwarded to the master.

## Setting conflict detection (update-anywhere replication)

**Default:** No conflict detection

### Restrictions:

- Tables from non-DB2 relational databases cannot participate in update-anywhere; therefore, non-DB2 relational sources do not have conflict detection.
- If you have an update-anywhere configuration that includes DATALINK columns, you must specify None for the conflict-detection level. DB2 does not check update conflicts for external files pointed to by DATALINK columns.
- If you have an update-anywhere configuration that includes LOB columns, you must specify None for the conflict-detection level. Columns of LOB data type cannot participate in update-anywhere replication.

In update-anywhere configurations, conflicts can sometimes occur between the master and its replicas. Conflicts can happen when:

- An update is made to a row in the master table and a different update is made to the same row in one or more replica tables, and the Apply program processes the conflicting changes during the same cycle.
- Constraints are violated.

Although you set the conflict-detection level for individual replication sources, the Apply program uses the highest conflict-detection level of any subscription-set member as the level for all members of the set.

DB2 replication provides three levels of conflict detection: no detection, standard detection, and enhanced detection. Based on your tolerance for lost or rejected transactions and performance requirements, you can decide which type of detection to use. You can select from the following levels of conflict detection when you register a source that you plan to use for update-anywhere replication:

**None** No conflict detection. Conflicting updates between the master table and the replica table will not be detected. This option is *not* recommended for update-anywhere replication.

#### **Standard**

Moderate conflict detection.

During each Apply cycle, the Apply program compares the key values in the master's CD table with those in the replica's CD table. If the same key value exists in both CD tables, it is a conflict. In case of a conflict, the Apply program will undo the transaction that was previously committed at the replica by reading from the replica's CD table and keeping only the changes that originated at the master.

#### **Enhanced**

Conflict detection that provides the best data integrity among the master and its replicas.

Like with standard detection, the Apply program compares the key values in the master's CD table with those in the replica's CD table during each Apply cycle. If the same key value exists in both CD tables, it is a conflict. However, with enhanced detection, the Apply program waits for all inflight transactions to commit before checking for conflicts. To ensure that it catches all inflight transactions, the Apply program locks all target tables in the subscription set against further transactions and begins conflict detection after all changes are captured in the CD table. In case of a conflict, the Apply program will undo the transaction that was previously committed at the replica by reading from the replica's CD and keeping only the changes that originated at the master.

**Restriction:** Even if you specify enhanced conflict detection, when the Apply program runs in an occasionally-connected environment (started with the COPYONCE keyword), the Apply program uses standard conflict detection.

The Apply program cannot detect read dependencies. If, for example, an application reads information that is subsequently removed (by a DELETE statement or by a rolled back transaction), the Apply program cannot detect the dependency.

If you set up a replication configuration where conflicts are possible (by selecting either no detection or standard detection), you should include a method for identifying and handling any conflicts that occur. Even though the replication infrastructure has detected and backed out transaction updates that were in conflict, the application designer must decide what to do about transactions that were at one time committed and now have been backed out. Because the ASNDONE exit routine runs at the end of each subscription cycle, the application designer can use it as a launching point for such application-specific logic. The information regarding conflicting updates that were backed out will remain in the CD and UOW tables until they are eligible for retention limit pruning. For details about how to add this application-specific logic, see “Modifying the ASNDONE exit routine (UNIX, Windows, z/OS)” on page 151 or “Modifying the ASNDONE exit routine (OS/400)” on page 152, depending on which platform you are using.

## Registering tables that use remote journaling (OS/400)

**Default:** remote journal is *not* used as the source

When registering OS/400 tables that use remote journaling, you can define for DB2 replication to use the remote journal as the replication source instead of the local journal. By selecting the remote journaling option for replication, you move the CD tables, the Capture program, and the Capture control tables to an OS/400 database server that is separate from the OS/400 server that the source table is on.

When you register tables on OS/400 as sources, the default assumes that you do *not* want to use remote journaling.

**Recommendation:** Whenever you are replicating data from one OS/400 table to another OS/400 table and you have a remote journal set up, it is highly recommended that you use the remote journaling function when registering. Using remote journaling in replication will greatly increase performance. Because the remote journal function makes it possible to move the registration, the Capture program, and the Capture control tables away from the system on which the source table resides, more resources are left available on that system. This reduces processor usage and saves disk space. Also,

when you use a remote journal that resides at the target server, the CD table is on the same system as the target table, which allows the Apply program to apply changes directly from the CD table to the target table without using a spill file. Not using a spill file reduces the amount of resources used by the Apply program.

**Suggestion:** Register tables that use remote journals as sources only if the registration resides on the same OS/400 system as the replication target. DB2 replication allows you to register remote journals as sources even if the registration does not reside on the same OS/400 system as the target, but then you don't get the performance advantages that you do from having the journal on the target system.

Before you register an OS/400 table that uses remote journaling, make sure that your remote journal is in an active state.

For more information about the remote journal function, see *Backup and Recovery*, SC41-5304, and *OS/400 Remote Journal Function for High Availability and Data Replication*, SG24-5189.

## Using relative record numbers (RRN) instead of primary keys (OS/400)

Typically, the target table for a source uses the same key columns as the primary key columns in the source. The Apply program uses this key value to track which data it has replicated from the source's CD table to the target. If you are registering an OS/400 table that does not have a primary key, a unique index, or a combination of columns that can be used as a unique index, you must register the table using the relative record numbers (RRN). When you choose to replicate using the RRN, both the CD table and the target table have an extra column, IBMQSQ\_RRN of type INTEGER, which contains a unique value for each row. This column contains the RRN that corresponds to each source table row.

The RRN is used as a primary key for the source table row as long as the source table is not reorganized. When the source table is reorganized, the RRN of each source table row changes; therefore, the RRN in the CD and target table rows no longer has the correct value that reflects the row's new position in the source table. Any time that you reorganize a source table (to compress deleted rows, for example), DB2 DataPropagator for iSeries performs a full refresh of all the target tables in the set of that source table. For this reason, place target tables that use RRN as primary keys in subscription sets with other targets that use RRNs, and not in sets with tables that use some other uniqueness factor.

---

## How views behave as replication sources

When you register views for replication, they inherit the registration options from the registration definitions of their underlying tables. Most importantly, the underlying tables of the view determine whether the view is registered for change-capture replication or full-refresh only. The following sections describe how registered views behave in replication in various scenarios:

- “Views over a single table”
- “Views over a join of two or more tables”

### Views over a single table

You can register a view over a single table if the underlying table is registered for replication. When you register a view over a single registered table, the view inherits the type of replication that the underlying table has. If the underlying table is registered for full-refresh-only replication, the view has full-refresh-only replication. You cannot register the view for change-capture replication because the underlying table does not have a CD table associated with it to keep track of changes. If the underlying table is registered for change-capture replication, the view has change-capture replication and cannot be registered for full-refresh only.

When you register a view over a table that is registered for change-capture replication, a view is created for you over the underlying tables’ CD table. This CD view contains only the columns referenced by the registered view.

You cannot register a subset of columns in the view; all of the columns in the view are automatically registered.

### Views over a join of two or more tables

When you register a view over a join of two or more tables, the underlying tables can be registered or non-registered tables, as long as at least one table in the join is registered. You can also have inner-joins of CCD tables that are registered as sources.

When you register a join as a replication source, DB2 replication adds multiple rows in the register (IBMSNAP\_REGISTER) table with identical SOURCE\_OWNER and SOURCE\_TABLE values. These rows are distinguished by their SOURCE\_VIEW\_QUAL values. Each of these entries identifies a component of the join.

**Restriction:** If you define a join that includes a CCD table, all other tables in that join must be CCD tables.

For a join view to be a viable replication source, you must create it using a correlation ID. (Views over single tables do not require a correlation ID.)



**Example:**

```
create view REGRES1.VW000 (c000,c1001,c2001,c2002,c1003) as
  select a.c000,a.c001,b.c001,b.c002,a.c003
  from REGRES1.SRC001 a, REGRES1.SRC005 b
  where a.c000=b.c000;
```

where VW000 is the name of the view. SRC001 and SRC005 are the tables that are part of the view and C000, C001, C002, and C003 are the columns that are part of the view under the condition that the C000 columns are equal in both tables (SRC001 and SRC005).

The type of replication that the view inherits depends on the combination of its underlying tables, each of which can be:

- Registered for change-capture replication
- Registered for full-refresh-only replication
- Not registered

Table 2 shows the various combinations of underlying tables and what type of source view and CD view results from each combination.

*Table 2. Combinations of underlying tables for views*

Table 1	Table 2	Description of join view and CD view
Registered for change capture	Registered for change capture	The view is registered for change-capture replication. The CD views contain the referenced columns from Table 1's CD table and from Table 2's CD table.
Registered for change capture	Registered for full-refresh only	The view is registered for change-capture replication. The CD view contains the referenced columns from Table 1's CD table and the referenced columns from Table 2. Only changes to columns that are in Table 1 will be replicated to the registered view's target during each replication cycle.
Registered for full-refresh only	Registered for full-refresh only	The view is registered for full-refresh-only replication. There is no CD view.
Registered for full-refresh only	Not registered	The view is registered for full-refresh-only replication. There is no CD view.
Registered for change capture	Not registered	The view is registered for change-capture replication. The CD view contains referenced columns from Table 1's CD table and the referenced columns from Table 2. Only changes to columns that are in Table 1 will be replicated to the registered view's target during each replication cycle.
Not registered	Not registered	The view is not a valid replication source and cannot be registered.

When you define a view that includes two or more source tables as a replication source, you must take care to avoid double deletes. A double-delete occurs when you delete a row during the same replication cycle from both tables that are part of a view. For example, suppose that you create a view that contains the CUSTOMERS table and the CONTRACTS table. A double-delete occurs if you delete a row from the CUSTOMERS table and also delete the corresponding row (from the join point of view) from the CONTRACTS table during the same replication cycle. The problem is that, because the row was deleted from the two source tables of the join, the row does not appear in the views (neither base views nor CD-table views), and thus the double-delete cannot be replicated to the target.

To avoid double-deletes, you must define a CCD table for one of the source tables in the join. This CCD table should be condensed and non-complete and should be located on the target server. Defining a condensed and non-complete CCD table for one of the source tables in the join solves the double-delete problem in most situations because the IBMSNAP\_OPERATION column in the CCD table allows you to detect the deletes. Simply add an SQL statement to the definition of the subscription set that should run *after* the subscription cycle. This SQL statement removes all the rows from the target table for which the IBMSNAP\_OPERATION is equal to “D” in the CCD table.

Problems with updates and deletes can still occur if, during the same Apply cycle, a row is updated on the source table that has the CCD while the corresponding row is deleted on the other table in the join. As a result, the Apply program is unable to find the corresponding row in the joined table and cannot replicate the updated value.

---

## Registering views of tables as sources

This section describes how to register views of DB2 tables as replication sources.

### Prerequisites:

- Capture control tables must already exist on the Capture control server that will process the view that you want to register as a source. If you need to create Capture control tables, see “Setting up the replication control tables” on page 23.
- The name of the source views must follow the DB2 table naming conventions.

### Restrictions:

- You cannot register views of non-DB2 relational tables.
- You cannot register a view that is over another view.

- On OS/400, because SQL statements are limited to a length of 32,000 characters, you can register only approximately 2000 columns per view; the exact number of columns depends on the length of the column names.
- All CCD tables that have views defined over them must be complete and condensed to be registered as a replication source.

**Recommendation:** Before you register the view as a source, register at least one of the underlying tables as a source. For information on how to register a table, see “Registering DB2 tables as sources” on page 37.

### **Procedure:**

Use one of the following methods to register a view:

#### **Replication Center**

Use the Register Views window. See the Replication Center help for details.

#### **System commands for replication (OS/400)**

Use the **ADDDPRREG** system command. See “ADDDPRREG: Adding a DPR registration (OS/400)” on page 349 for the syntax of this command and a description of its parameters.

When you register a view, you identify which view you want to register by specifying the name of the view, the source server, and the Capture schema. Registration options for views are derived from the registration definition of the source tables on which the views are defined. See “Registration options for source tables” on page 41 for a complete list of registration options, their defaults, and an explanation of when you might want to use or change the defaults. For information on what type of replication that the view will inherit based on its underlying tables (change-capture or full-refresh only), see “How views behave as replication sources” on page 58.

---

## **Maintaining CCD tables as sources (IMS)**

If you have externally populated CCD tables that are not populated by the Apply program or maintained by a program such as IMS DataPropagator or DataRefresher, you must maintain these tables so that the Apply program can read the CCD tables as sources or function correctly. This section describes how to maintain CCD tables as replication sources.

To maintain a CCD table that is populated by an external tool, you must update three columns in the register (IBMSNAP\_REGISTER) table: **CCD\_OLD\_SYNCHPOINT**, **SYNCHPOINT**, and **SYNCHTIME**. (For details about these columns in the register table, see “*schema*.IBMSNAP\_REGISTER” on page 498.) You must update these three columns for each of the following types of events:

- On an initial full refresh or load of the CCD table:
  - Set `CCD_OLD_SYNCHPOINT` to a value that represents the minimum value of `IBMSNAP_COMMITSEQ` from the CCD table.
  - Set `SYNCHPOINT` to a value that represents the maximum value of `IBMSNAP_COMMITSEQ` from the CCD table. Do not set `SYNCHPOINT` to 0. If you are creating your own values for sequencing, start with a `SYNCHPOINT` value of 1.
  - Set `SYNCHTIME` to a value that represents the maximum timestamp value of `IBMSNAP_LOGMARKER` from the CCD table.
- On any update to the CCD table after the full refresh or load:
  - Do not change the `CCD_OLD_SYNCHPOINT` value.
  - Set `SYNCHPOINT` to a value that represents the new maximum value of `IBMSNAP_COMMITSEQ` from the CCD table.
  - Set `SYNCHTIME` to a value that represents the new maximum timestamp value of `IBMSNAP_LOGMARKER` from the CCD table.
- On any subsequent full refresh or load of the CCD table:
  - Set `CCD_OLD_SYNCHPOINT` to a value that represents the minimum value of `IBMSNAP_COMMITSEQ` from the CCD table.
  - Set `SYNCHPOINT` to a value that represents the maximum value of `IBMSNAP_COMMITSEQ` from the CCD table.
  - Set `SYNCHTIME` to a value that represents the maximum timestamp value of `IBMSNAP_LOGMARKER` from the CCD table.

**Important:** This assumes that the values that are used in the CCD table for `IBMSNAP_COMMITSEQ` and `IBMSNAP_LOGMARKER` are always increasing values. The Apply program will not detect that a full refresh has been performed on the source CCD table unless the `CCD_OLD_SYNCHPOINT` value is larger than the most recently applied `SYNCHPOINT` value.

**Related concepts:**

- Chapter 14, “Using the DB2 Replication Center” on page 243

**Related tasks:**

- Chapter 4, “Subscribing to sources” on page 63

**Related reference:**

- “ADDDPRREG: Adding a DPR registration (OS/400)” on page 349

---

## Chapter 4. Subscribing to sources

After you register the tables and views that you want to use as replication sources, you can define a subscription for your target tables or views so that they receive the source data and the changes from those sources. The administration tasks described in this chapter help you set up the control information that the Capture and Apply programs use to copy source data or to capture changed data and replicate it to the target tables at the appropriate interval.

This chapter consists of the following sections:

- “Planning how to group sources and targets”
- “Creating subscription sets” on page 66
- “Processing options for subscription sets” on page 68
- “Mapping source tables and views to target tables and views within a subscription set” on page 75
- “Selecting a target type” on page 78
- “Common properties for all target table types” on page 90

---

### Planning how to group sources and targets

Before you define which targets subscribe to which sources, you need to plan how you want to group your sources and targets. DB2 replication processes source-to-target mappings in groups. These groups consist of one or more sources that are processed by the same Capture program and one or more targets that subscribe to all or part of the source data, which are processed by the same Apply program. These groups are called *subscription sets*, and the source-to-target mappings are called *subscription-set members*.

When planning for subscription sets, be aware of the following rules and constraints:

- A subscription set maps a source server with a target server. A subscription-set member maps a source table or view with a target table or view. Subscription sets and subscription-set members are stored in the Apply control server.
- The Apply program processes all members in a subscription set as a single group. Because of this, if any member of the subscription set requires full-refresh copying for any reason, all members for the entire set are refreshed.

- All source tables in the members of a set must have the same Capture schema.
- On OS/400 systems, all source tables in the members of a subscription set must be journaled to the same journal.
- All external CCD tables created by IMS DataPropagator that are members of a subscription set must have the same Capture schema.

A single Apply program, which has a unique Apply qualifier, can process one or many subscription sets. A single subscription set can contain one or many subscription-set members. The following sections discuss the trade-offs for having few or many sets per Apply program, and few or many subscription-set members per subscription set.

### **Planning the number of subscription-set members**

When you add members to a subscription set, you must decide whether to group all of your source-target pairs (subscription-set members) into one subscription set, create separate subscription sets for each pair, or create a small number of subscription sets, each with a number of pairs.

Because the Apply program replicates the members of a subscription set in one (logical) transaction, you should group multiple members into one subscription set in either of the following situations:

- If the source tables are logically related to one another.
- If the target tables have referential integrity constraints.

By grouping multiple members into one subscription set, you can ensure that replication for all members begins at the same time. Also, you reduce the number of database connections needed to process the subscription sets and you reduce the administration overhead for maintaining your replication environment. If the subscription set contains SQL statements or stored procedures, you can use those statements or procedures to process all of the members of the subscription set.

If there are no logical or referential integrity relationship between the tables in a subscription set, you can group them into one subscription set or into several subscription sets. The main reason for limiting the number of subscription sets is to make administration of the replication environment simpler. But by increasing the number of subscription sets, you minimize the affect of replication failures.

If you want to be able to more easily locate any errors that cause the Apply program to fail, add only a small number of members to a subscription set. With fewer members, you will likely find the source of the problem more quickly than if the set contains a large number of members. If one member of a subscription set fails, all of the data that has been applied to other members

of the set is rolled back; so that no member can complete the cycle successfully unless all members do. The Apply program rolls back a failed subscription set to its last successful commit point, which could be within the current Apply cycle if you specified the **commit\_count** keyword when you started the Apply program.

### **Planning the number of subscription sets per Apply qualifier**

When you define a subscription set, you specify the Apply qualifier for that subscription set. The Apply qualifier associates an instance of the Apply program with one or more subscription sets. Each subscription set is processed by only one Apply program, but each Apply program can process one or more subscription sets during each Apply cycle.

You can run as many instances of the Apply program (each with its own Apply qualifier) as you need, and each Apply program can process as many subscription sets as you need. You have two basic options:

- Associate each Apply qualifier with one subscription set (each Apply program processes exactly one subscription set)  
If speed is important, you can spread your sets among several Apply qualifiers, which allows you to run several instances of the Apply program at the same time. If you decide to have an Apply program process one subscription set, you can use the Apply program's OPT4ONE startup option, which loads the control-table information for the subscription set into memory. Using this option, the Apply program does not read the control tables for the subscription-set information for every Apply cycle. Therefore, the Apply program performs better. However, the more Apply programs that you run, the more system resources they will use, and the slower their overall performance might be.
- Associate each Apply qualifier with multiple subscription sets (each Apply program processes many subscription sets)

By using more than one Apply qualifier, you can run more than one instance of the Apply program from a single user ID.

The Apply program tries to keep all sets for a given Apply qualifier as current as possible. When an Apply cycle starts, the Apply program determines which of the subscription sets contains the least current data and starts processing that set first.

If speed is not your main goal, you might want to replicate a large number of subscription sets with one Apply qualifier. For example, this could be a very good option if you wait until after business hours before replicating.

One disadvantage of having one Apply program process multiple subscription sets is that the Apply program processes the subscription sets sequentially; thus, your overall replication latency can increase.

If you have specific requirements for certain subscription sets, you can combine these two options. For example, you could have one Apply program process most of your subscription sets and thus take advantage of using one Apply program to process related subscription sets together, and you can have another Apply program process a single subscription set and thus ensure minimum replication latency for that subscription set. And by using two instances of the Apply program, you increase the overall parallelism for your subscription sets.

---

## Creating subscription sets

Before you replicate data from a registered source, you must create subscription sets, which are collections of subscription-set members (source-to-target mappings) that the Apply program processes as a set. This section discusses the properties that you define for each subscription set. These properties, which apply to every member that you add to the set, define which servers you want to replicate data to and from, including which Capture program (based on the Capture schema for the registered source) and Apply programs you want to use, and when and how you want the Apply program to process the set.

You don't have to add subscription-set members to a subscription set; instead, you can create an empty set, which is a set that doesn't contain any source-to-target mappings. You might want to create an empty set for the following reasons:

- You plan to add members to a set later and don't plan to activate the subscription set until you add members.
- You want the Apply program to process the empty subscription set in order to call an SQL statement or a stored procedure whenever the set is eligible for processing.

### Prerequisites:

1. You must create the Apply control tables in the Apply control server for the subscription set.
2. Before you add subscription-set members to subscription sets, you must register the tables or views that you want to use as sources. If you need to register sources for replication, read and follow the instructions in Chapter 3, "Registering tables and views as replication sources" on page 37. You should also consider how you want to group your sets. If you need to plan for your sets, see "Planning how to group sources and targets" on page 63 for more information.

### Procedure:

To create a subscription set, you can use either of the following two methods:



**Replication Center**

Use the Create subscription sets notebook. See the Replication Center help for details.

**System commands for replication (OS/400)**

Use the **ADDDPRSUB** system command. See “ADDDPRSUB: Adding a DPR subscription set (OS/400)” on page 359 for the syntax of this command and a description of its parameters.

To create a subscription set, you provide these basic characteristics:

**Apply control server alias**

The local alias of the server containing the control tables for the Apply program that will process the subscription set. Define the same alias for the Apply control server in every database from which you run the Replication Center or the Apply program so that the Replication Center will populate the Apply control tables correctly and so that every Apply program will connect to the correct server using a standard alias name.

**Subscription set name**

The name of the subscription set. At the Apply control server that processes this subscription set, the set name must be unique for a given Apply qualifier. The name can be up to 18 characters long.

**Apply qualifier**

The name of a new or existing Apply qualifier, which identifies which Apply program will process this subscription set. You can use the same Apply qualifier to process multiple subscription sets. Subscription sets that have the same Apply qualifier must be defined in the same Apply control server. If you are creating a new Apply qualifier, see Chapter 16, “Naming rules for replication objects” on page 301 for the rules on how to name an Apply qualifier.

**Capture control server alias**

The alias of the server containing the control tables for the Capture program that will process the registered sources for the subscription set. Define the same alias for the Capture control server in every database from which you run the Replication Center or the Apply program so that the Replication Center will populate the Capture and Apply control tables correctly and so that every Apply program will connect to the correct server using a standard alias name.

**Capture schema**

The name of the Capture schema that identifies the set of Capture control tables that define the registered sources for the subscription

set. All of the source tables in a subscription set must reside on the same server, and only one Capture program can be capturing the changes for them.

### **Target server alias**

The name of the target server that contains the tables or views to which the Apply program will replicate changes from the source. Define the same alias for the target server in every database from which you run the Replication Center or the Apply program so that the Replication Center will populate the Apply control tables correctly and so that every Apply program will connect to the correct server using a standard alias name.

When you create a subscription set, you can use the default settings for how the Apply program processes the set, or you can modify the subscription properties to meet your replication needs. See “Processing options for subscription sets” for a complete list of processing options for subscription sets, their defaults, and an explanation of when you might want to use or change the defaults.

---

## **Processing options for subscription sets**

This section discusses the properties that you can define to specify how the Apply program process the subscription set. In addition, this section helps you to decide which settings to select based on your replication needs:

- “Specifying whether the set is active”
- “Specifying how many minutes worth of data the Apply program retrieves” on page 69
- “Specifying how the Apply program replicates changes for members in the set” on page 72
- “Defining SQL statements or stored procedures for the subscription set” on page 73
- “Scheduling the replication of a subscription set” on page 73

### **Specifying whether the set is active**

**Default:** Inactive

You can specify whether you want the Apply program to begin processing the subscription set. When you activate a subscription set, the Apply program initiates a full refresh for that set. You have three activation levels to choose from:

**Active** The Apply program processes the set during its next cycle. Activate the set if you want the Apply program to process the set the next time

it runs; you can still add members to the set later. When you activate the set, it remains active and the Apply program continues to process it until you deactivate it.

#### **Inactive**

The Apply program does not process the set. Leave the set inactive if you are not ready for the Apply program to process it.

#### **Active only once**

The Apply program processes the set during its next cycle and then deactivates the set. Specify this option if you want the set to run only once. Make sure that you add all the subscription-set members before selecting this option because the Apply program will not process members that you add later, unless you reactivate the subscription set.

### **Specifying how many minutes worth of data the Apply program retrieves**

**Default:** 20 minutes

You can specify an approximate number of minutes' worth of data for the Apply program to retrieve from the replication source during each Apply cycle. There are several situations for which this specification can be useful:

- When the amount of data to be processed within one subscription-set cycle is large.  
Subscription sets that replicate large blocks of changes in one Apply cycle can cause the spill files or logs (for the target database) to overflow. For example, batch-Apply scenarios can produce a large backlog of enqueued transactions that need to be replicated.
- An extended outage of the network can cause a large block of data to accumulate in the CD tables, which can cause the Apply program's spill file and the target's log to overflow.

The number of minutes that you specify is called the data block. The data-blocking value that you specify is stored in the MAX\_SYNC\_MINUTES column of the subscription sets (IBMSNAP\_SUBS\_SET) table. If the accumulation of data is greater than the size of the data block, then the Apply program converts a single Apply cycle into several mini-cycles. If resources are still not sufficient to handle the blocking factor provided, the Apply program reduces the size of the data block to match available system resources. By retrieving smaller sets of data, the Apply program can lessen both the network load and the temporary space required for the retrieved data.

**Example:** If you specify that the Apply program should retrieve at most 10 minutes' worth of data per mini-cycle, the Apply program will retrieve an amount of committed data from the CD table at the source that is within approximately 10 minutes of the last mini-cycle.

In addition to preventing the logs and spill files from overflowing, these mini-cycles have several other benefits. If there is an error during the replication cycle, the Apply program must roll back only the changes that it made during the mini-cycle that failed. If replication fails during a mini-cycle, the Apply program tries to process the subscription set from the last successful mini-cycle, which can save a significant amount of time if a large amount of changed data is available to be processed. Figure 5 shows how the changed data is broken down into subsets of changes.

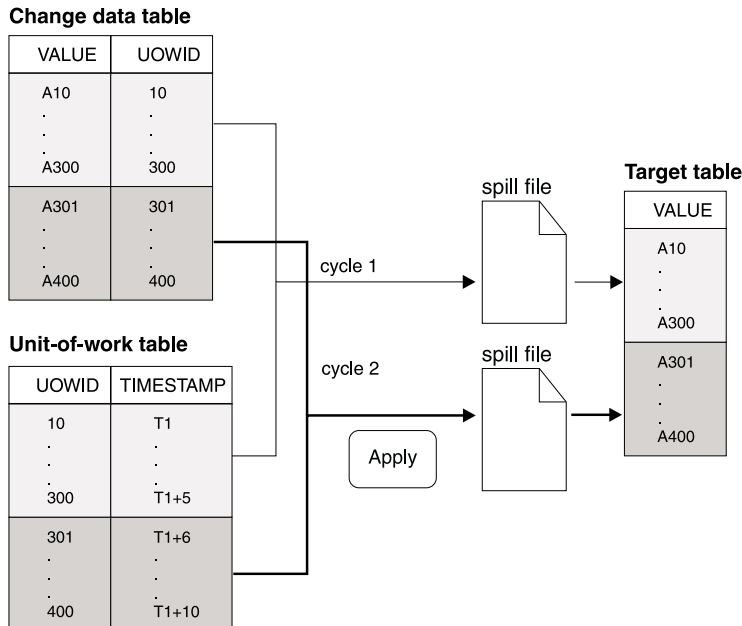


Figure 5. Data blocking. You can reduce the amount of network traffic by specifying a data-blocking value.

The number of minutes that you set should be small enough so that all transactions for the subscription set that occur during the interval can be copied without causing the spill files or log to overflow during the mini-cycle.

**Important for OS/400:** Ensure that the total amount of data to be replicated during each Apply cycle does not exceed 4 million rows.

When processing data, the Apply program does not take any of the following actions:

- Split a unit of work (meaning that a long running batch job without commits cannot be broken up by the data blocking factor).
- Roll back previously committed mini-subscription cycles.
- Use the data blocking factor during a full refresh.

## Deciding how the Apply program loads target tables that have referential integrity

If you want to have referential integrity between target tables in a set, you must choose how the Apply program will process the source data during the initial load of the target tables. By default, the Apply program performs the full refresh of the targets by reading all the rows in the source, storing them in memory, and then inserting the rows into the targets tables. However, you have other options about the way that the target tables are initially loaded. You do not make these decisions when you create subscription sets or define the source-to-target mappings (members) for each set; you decide how the targets will be loaded when you set the startup parameters for the Apply program. You should consider your replication requirements for each subscription-set member while you are defining it and can decide which startup option you will use for the Apply program that will process each member.

Consider one of the following two alternatives on when you can create these referential integrity relationships:

- Before the target tables are populated.  
This requires that no changes are made at the source table during the entire extract and load stage of the target table. Also, you must start the Apply program using the LOADX startup option in order to get the speed of the load processing and to bypass the referential constraint checking during this initial population. If you do not use the Apply startup option of LOADX, the inserts in the target table could fail.
- After the Apply program has fully populated the target tables and has processed one complete and successful cycle of applying changes to this set of tables.

The advantage waiting to add the referential integrity constraints to these tables is that the changes can still be made at the source table while the target tables are being loaded. You can start the Apply program with or without the LOADX startup option, because there are no constraints that need to be bypassed. A full refresh will typically be much faster using the LOADX startup option. During the initial population of the target tables, the targets might be out of synch with each other regarding their referential integrity relationships; but, as they are being loaded, all changes are being captured for the set. After the Apply program replicates the first set of changes, all target tables will contain the same transactions and will have referential integrity. At this point, you can deactivate the set, add the referential integrity constraints, and then reactivate the set.

For more information about the startup options that you have for how your target tables are initially loaded, see “Refreshing target tables using the ASNLOAD exit routine” on page 154.

## Specifying how the Apply program replicates changes for members in the set

When the subscription set has change-capture replication, you can decide how you want the Apply program to replicate changes to every source-to-target mapping in the set. After the target tables are initially loaded, the Apply program starts to read the CD (or CCD) tables and collects the changes into spill files. For each CD (or CCD) table, the Apply program creates a separate spill file. The Apply program then reads the changes from the spill files and applies them to the target tables. It can do this in one of three ways:

- Using table-mode processing
- Using transaction-mode processing
- Using a mixture of table-mode and transaction-mode processing, depending on the target-table types in the subscription set

By specifying the type of processing for the subscription set, you can control how often the Apply program commits its changes to the target table or view. The Apply program can commit once for each subscription-set member or after applying a number of transactions. Having one commit can reduce the latency for the subscription set, but having multiple commits allows the Apply program to apply the data in the original commit sequence.

### Table mode

The Apply program reads all changes from a spill file for a CD (or CCD) table, applies the changes to the corresponding target tables, and then begins to process the spill file for the next CD (or CCD) table. When it is done reading and applying changes from all the CD (or CCD) tables in the set, it then issues a DB2 commit to commit all of the changes to all of the target tables in the subscription set.

### Transaction mode

The Apply program opens all of the spill files at once and processes the changes from them at the same time. The order in which the changes are applied to the target tables is the order in which the transactions took place at the source tables. The Apply program issues a DB2 commit at intervals that you specify when you start the Apply program. Use this type of processing when you have referential integrity constraints on the target tables in the subscription set.

You can specify that the Apply program use transaction-mode processing for any subscription set, however, it only changes the Apply program's behavior for sets with user-copy and point-in-time target tables, but not sets with the following types of target tables:

- CCD target tables. Sets containing CCD tables as sources are always processed in table mode.

- Any target table for which the source table is a CCD table. Sets containing CCD tables are always processed in table mode.
- Replica target tables. Sets containing replica tables are always processed in transaction mode.

## Defining SQL statements or stored procedures for the subscription set

You can define SQL statements or stored procedures that run each time the Apply program processes the subscription set. These statements can be useful for pruning CCD tables or manipulating source data before it is applied to targets. You can specify when and where the SQL statements or stored procedures should run:

- At the Capture control server before the Apply program applies the data.
- At the target server before the Apply program applies the data.
- At the target server after the Apply program applies the data.

The Apply program processes the statements or procedures in the order listed above.

When you use the Replication Center to add SQL statements to a subscription set, you can click **Prepare statement** in the Add SQL Statement or Procedure Call window to verify the statement's syntax.

## Scheduling the replication of a subscription set

You can control how often the Apply program processes a subscription set and thereby control how current the data in your target tables is. You can control how often the subscription set is eligible for processing by using time-based or event-based scheduling, or you can use these scheduling options together. The Apply program begins processing a subscription set when the subscription set is eligible for processing. For example, you can set an interval of one day, and also specify an event that triggers the subscription cycle. If you use both of these scheduling options, the subscription set will be eligible for processing at both the scheduled time and when the event occurs.

If there is a large amount of data to be replicated during an interval or between events for any subscription set that the Apply program processes, it is possible for a particular subscription set to become eligible for processing, but the Apply program will not be able to process it until it finishes applying data for all subscription sets in the prior interval or for the prior event. As soon as it finishes processing the subscription set, the Apply program begins processing the next eligible subscription set. In this case, you might not get the expected replication latency, but you won't lose any data.

### Time-based scheduling

The simplest method of controlling when the set is processed is to use time-based scheduling (also known as relative timing or interval timing). You determine a specific start date, time, and interval. The interval can be specific

(from one minute to one year) or continuous, but time intervals are approximate. The Apply program begins processing a subscription set as soon as it is able, based on its workload and the availability of resources. Choosing a timing interval does not guarantee that the frequency of replication will be exactly at that interval. If you specify continuous timing, the Apply program replicates data as frequently as it is able.

### Event-based scheduling

To replicate data using event-based scheduling (also known as event timing), you can specify an event name when you define the subscription set. To allow the Apply program to recognize the event when it occurs, you must also populate the subscription events (IBMSNAP\_SUBS\_EVENT) table with a timestamp for the event name. When the Apply program detects the event, it begins replication.

The subscription events table has four columns, as shown in Table 3.

*Table 3. The subscription events table*

EVENT_NAME	EVENT_TIME	END_OF_PERIOD	END_SYNCHPOINT
END_OF_DAY	2002-05-01- 17.00.00.000000	2002-05-01- 15.00.00.000000	

EVENT\_NAME is the name of the event that you specify while defining the subscription set. EVENT\_TIME is the timestamp for when the Apply program begins to process the set. END\_OF\_PERIOD is an optional value that indicates that updates that occur after the specified time should be deferred until a future event or time. END\_SYNCHPOINT is also an optional value that indicates that updates that occur after the specified log-sequence number should be deferred until a future event or time. If you specify values for both END\_OF\_PERIOD and END\_SYNCHPOINT, the value for END\_SYNCHPOINT takes precedence. Set the EVENT\_TIME value using the clock at the Apply control server, and set the END\_OF\_PERIOD value using the clock at the source server. This distinction is important if the two servers are in different time zones.

In Table 3, for the event named END\_OF\_DAY, the timestamp value for EVENT\_TIME (2002-05-01-17.00.00.000000) is the time when the Apply program should begin processing the subscription set. The END\_OF\_PERIOD timestamp value (2000-05-01-15.00.00.000000) is the time after which updates are not replicated and will be replicated on the next day's cycle. That is, the event replicates all outstanding updates made before three o'clock, and defers all subsequent updates.

You or your applications must post events to the subscription events (IBMSNAP\_SUBS\_EVENT) table using an SQL INSERT statement to insert a



row into the table to activate the event. For example, use the current timestamp plus one minute to trigger the event named by `EVENT_NAME`. Any subscription set tied to this event becomes eligible to run in one minute. You must manually post events for both full refresh and change-capture replication.

You can post events in advance, such as next week, next year, or every Saturday. If the Apply program is running, it starts at approximately the time that you specify. If the Apply program is stopped at the time that you specify, when it restarts, it checks the subscription events table and begins processing the subscription set for the posted event.

The Apply program does not prune the table; you must populate and maintain this table. Also, you cannot use the Replication Center to update the subscription events table. You must issue SQL statements or define automated procedures to add events to this table.

**Example:**

```
INSERT INTO ASN.IBMSNAP_SUBS_EVENT
    (EVENT_NAME, EVENT_TIME)
VALUES ('EVENT01', CURRENT_TIMESTAMP + 1 MINUTES)
```

Any event that occurs prior to the most recent time that the Apply program processed the subscription set (as specified by the value in the `LASTRUN` column of the subscription-set control table) is considered to be an expired event and is ignored. Therefore, if the Apply program is running, you should post events that are slightly in the future to avoid posting an expired event.

---

## Mapping source tables and views to target tables and views within a subscription set

Within a subscription set, you can add source-to-target mappings that the Apply program will process as a group when it processes the set. These source-to-target mappings are called subscription-set members. When defining a subscription-set member, you specify which target table or view subscribes to the source data, and you can define how you want the replicated data to appear at the target.

**Prerequisites:**

Before you set up targets that subscribe to changes at sources, you must register the tables or views that you want to use as sources. If you didn't already register sources for replication, read and follow the instructions in Chapter 3, "Registering tables and views as replication sources" on page 37. You should also create a subscription set and plan for how many members you want to add in a set. If you need to create a subscription set, see

“Creating subscription sets” on page 66. If you need to plan for subscription-set members, see “Planning the number of subscription-set members” on page 64.

**Restrictions:**

- DB2 replication does not support views of non-DB2 relational tables as sources.
- If you define a target view, that view must be an insertable view. That is, all of the view's columns must be updateable and the full select for the view cannot include the keywords UNION ALL.
- If you are using the Replication Center, you cannot add a column to a subscription-set member if that column does not already exist in the target table.
- **For Windows, UNIX, z/OS:** You can define a maximum of 200 members for each subscription set.
- **For OS/400:** You can define a maximum of 78 members for each subscription set.

**Procedure:**

To add a subscription-set member, you can use either of the following two methods:

**Replication Center**

Use one of the following notebooks:

- **Create Subscription Set.** Use this notebook when you create the subscription set.
- **Subscription Set Properties.** Use this notebook if you have already created the subscription set and want to add one or more subscription-set members to it.
- **Add Members to Subscription Sets.** Use this notebook to add one member to multiple subscription sets. For example, if you select four subscription sets when you open this notebook, you can add one member to each. Each member must use the same source.

See the Replication Center online help for details.

**System commands for replication (OS/400)**

Use the **ADDDPRSUBM** system command. See “ADDDPRSUBM: Adding a DPR subscription-set member (OS/400)” on page 376 for the syntax of this command and a description of its parameters.

To map a source with a target, specify the following information about the registered table or view that you want to use as the source:

- The source table or view and a target table or view (including a table space and index for the target table).
- The type of target table.
- The registered columns from the source table that you want to replicate to the target table.

When you use the Replication Center to map a source with a target, LOB columns and DATALINK columns are not automatically included in the column mapping. You must explicitly select those columns.

- The rows from the source table that you want to replicate to the target table (you include a WHERE clause to specify the rows).

To map the chosen source to a DB2 target, specify the following information about the target table or view:

- The schema of the target table or view.
- The name of the table or view you want to use as the target.

**Default:** The default name comes from the target object profile for the target server, if there is one. If you have not set this profile, the default is TG followed by the name of the source table or view. (For example, if the name of your source table is EMPLOYEE, the name of your target table defaults to TGEMPLOYEE.)

- The type of target table

**Default:** user copy

If the specified target table does not exist, the Replication Center or the **ADDPRSUBM** system command creates it.

To map the chosen source to a non-DB2 relational target, specify the following information about the target table:

- The schema of the nickname of the target table
- The nickname of the target table
- The remote schema
- The name of the remote table

**Default:** The default name comes from the target object profile for the target server, if there is one. If you have not set this profile, the default is TG followed by the name of the source table or view. (For example, if the name of your source table is EMPLOYEE, the name of your target table defaults to TGEMPLOYEE.)

- The type of target table

**Default:** user copy

When you add a subscription-set member, you can use the default target table type of user copy, or you can select another target table type to meet your replication needs.

When you add a subscription-set member for a target table that does not yet exist, you can use the default settings, or you can modify the member properties to meet your replication needs. You can first pick the type of target table that you want to use, and you can then set properties for how the Apply program replicates data to that target. See “Selecting a target type” for a description of various replication scenarios and which target table types you might want to use in each case. The section also helps guide you through which settings to choose based on your replication goals. Regardless of what target type you select, you can modify the common set of properties that all members share. See “Common properties for all target table types” on page 90 for a complete list of options for subscription-set members, their defaults, and an explanation of when you might want to use or change the defaults.

---

## Selecting a target type

This section gives you a description of each of the types of target tables you can choose from, and then it helps you decide which type of target table to select and how to define the target-table properties based on your replication goals. It also discusses what you should do if you want to use an existing table as your target. The type of target table that you need depends on how you want your data to appear at the target and what replication configuration you have. You can use either an existing table as your target, or you can create a new table.

The names of all non-DB2 relational target tables and indexes must follow the DB2 table and index naming conventions.

### Restrictions:

- The null attributes of after-image target columns must be compatible with the null attributes for those columns of the source table or view. Use the SQL COALESCE expression to provide compatibility with existing columns.
- For source tables on non-DB2 relational databases, you can define only the following types of target tables:
  - User copy tables
  - Point-in-time tables
  - External CCD tables
- For source tables on OS/400 systems that use RRN columns as their key columns, you can define only the following types of target tables:
  - Point-in-time tables
  - External CCD tables

- For source tables in a z/OS subsystem, the encoding scheme for the CD and UOW tables must be the same if the Apply program will join these tables to satisfy a subscription-set WHERE clause for a user-copy table. See Appendix A, “UNICODE and ASCII encoding schemes on z/OS” on page 641 for more information about encoding schemes.

You can select from the following types of target tables:

#### **User copy**

Read-only target table that includes only those columns defined in the subscription-set member. A user-copy table can have the same structure as the source table or it can have a subset of source columns, with or without before images or calculated columns.

#### **Point-in-time**

Read-only target table that includes the columns defined in the subscription-set member and a timestamp column. A point-in-time table can have the same structure as the source table or it can have a subset of source columns, with or without before images or calculated columns.

#### **Base aggregate**

Read-only target table that uses SQL column functions (such as SUM and AVG) to compute summaries of the entire contents of the source table.

A base-aggregate table summarizes the contents of a source table. A base-aggregate table also includes a timestamp of when the Apply program performed the aggregation. Use a base-aggregate table to track the state of a source table on a regular basis.

#### **Change aggregate**

Read-only target table that uses SQL column functions (such as SUM and AVG) to compute summaries of the entire contents of recent changes made to the source table, which are stored in the CD table or in an internal CCD table.

A change-aggregate table summarizes the contents of a CD table or in an internal CCD table, rather than the source table. A change-aggregate table also includes two timestamps to mark the time interval for when the changes were captured (written to the CD or CCD table). Use a change-aggregate table to track the changes (UPDATE, INSERT, and DELETE operations) made between replication cycles.

#### **CCD (consistent-change data)**

Read-only target table with additional columns for replication control information. These columns include: a log-record number (or journal-record number), an indicator of whether the source table was

changed using an SQL INSERT, DELETE, or UPDATE statement, and the log record number and timestamp of the commit statement associated with the insert, delete, or update. You can also optionally include before-image columns and columns from the UOW table.

### **Replica**

Read/write target table for update-anywhere replication. A replica table is the only type of target table that your application programs and users can update directly. Thus, a replica table receives changes from the master table and from local application programs or users. Replica tables can have the same structure as the source table or they can have a subset of source columns, but they do not include any additional replication control columns (such as timestamps). Replica tables are supported only for DB2 databases.

The following sections discuss possible uses for each target type. Each section guides you through the types of target tables that you can use and how you can set the target-table properties to meet your replication needs:

- “Defining read-only target tables”
- “Replicating the net change for a row to the target table” on page 83
- “Defining middle tiers in a multi-tier configuration” on page 85
- “Defining read-write targets (update-anywhere)” on page 87
- “Using an existing table as the target table” on page 89

After you select your target table type, you can use the default settings for the target table, or you can modify the target table properties to meet your replication needs. See “Common properties for all target table types” on page 90 for a complete list of common target table options, their defaults, and an explanation of when you might want to use or change the defaults.

### **Defining read-only target tables**

**Target table types:** User copy, point-in-time, base aggregate, change aggregate, CCD

Depending on how you want the source data to appear at your target, you can define read-only target tables to contain:

- “A copy of the source table or view”
- “A history of changes or audit information” on page 82
- “A computed summary of data or changes at the source” on page 81

#### **A copy of the source table or view**

**Target table types:** User copy, point-in-time

**Copy of source table:** By default, a user copy table will be created as your target type when you define a subscription-set member. Use this default type

if you want the target table to match the source table at the time the copy is made. User copy tables do not contain any additional replication-control columns, but they can contain a subset of the rows or columns in the source table or additional columns that are not replicated.

**Copy of source table with timestamp:** Select point-in-time as your target type if you want to keep track of the time at which changes were applied to the target. A point-in-time target contains the same data as your source table, with an additional timestamp column added to let you know when the Apply program committed each row to the target. The timestamp column is originally null. Point-in-time tables can contain a subset of the rows or columns in the source table or additional columns that are not replicated.

### **A computed summary of data or changes at the source**

**Target table types:** Base aggregate, change aggregate

**Restrictions:** Non-DB2 relational targets cannot be aggregate target-table types. Non-DB2 relational sources cannot have aggregate target-table types.

You can create target tables that contain summaries of the entire contents of the source tables or of the most recent changes made to the source table data. For aggregate target-table types, you can define target columns by using aggregate SQL column functions such as COUNT, SUM, MIN, MAX, and AVG. These columns do not contain the original source data; they contain the computed values of the SQL function that you define. The Apply program doesn't create aggregations during full refresh; rows are appended over time as the Apply program processes the set. An advantage of using an aggregate table is that DB2 replication can replicate summary information only rather than each individual row, thus saving both network bandwidth and space in the target table.

### **Summarizing contents of the source table:**

Use a base-aggregate target table to track the state of a source table during each replication cycle. For a base-aggregate target table, the Apply program aggregates (reads and performs calculations) from the source table. A base-aggregate table also includes a timestamp of when the Apply program performed the aggregation.

If a registered source table has only a base-aggregate table as its target, you do not need to capture changes for the source table.

**Example:** Suppose that you want to know the average number of customers that you have each week. If your source table has a row for each customer, the Apply program would calculate the sum of the number of rows in your source table on a weekly basis and store the results in a base aggregate table.

If you perform the aggregation every week, the target table will have 52 entries that show the number of customers you had for each week for the year.

### **Summarizing contents of the CD or CCD table:**

Use a change-aggregate target table to track the changes (UPDATE, INSERT, and DELETE operations) made between replication cycles at the source table. For a change-aggregate target table, the Apply program aggregates (reads and performs calculations) from the CD or internal CCD table. A change-aggregate table also includes two timestamps to mark the time interval for when the Capture program inserted changes into the CD or CCD table.

**Example:** Suppose that you want to know how many new customers you gained each week (INSERTs) and how many existing customers you lost (DELETEs). You would count the number of inserted rows and deleted rows in the CD table on a weekly basis and store that number in a change-aggregate table.

**Important:** If the source table for a subscription-set member is registered for full-refresh only replication, then you cannot have a change aggregate target table, which requires a CD or CCD table at the source.

### **A history of changes or audit information**

#### **Target table type: CCD**

You might want to audit the source data or keep a history how the data is used. By using a CCD table as your target type, you can track the history of source changes in various ways, depending on how you define the CCD table. For example, you can track before and after comparisons of the data, when changes occurred, and which user ID made the update to the source table.

To define a read-only target table that keeps a history of your source table, define the target CCD table to include the following attributes:

#### **Noncondensed**

To keep a record of all of the source changes, define the CCD table to be noncondensed, so it stores one row for every change that occurs. Because noncondensed tables contain multiple rows with the same key value, do *not* define a unique index. A noncondensed CCD table holds one row per UPDATE, INSERT, or DELETE operation, thus maintaining a history of the operations performed on the source table. If you capture UPDATE operations as INSERT and DELETE operations (for partitioning key columns), the CCD table will have two rows for each update, a row for the DELETE and a row for the INSERT.



### **Complete or noncomplete**

You can choose whether you want the CCD table to be complete or noncomplete. Because noncomplete CCD tables do not contain a complete set of source rows initially, create a noncomplete CCD table to keep a history of updates to a source table (the updates since the Apply program began to populate the CCD table).

### **Include UOW columns**

For improved auditing capability, include the extra columns from the UOW table. If you need more user-oriented identification, columns for the DB2 for z/OS correlation ID and primary authorization ID or the OS/400 job name and user profile are available in the UOW table. For details on which UOW columns you can include in a CCD table, see “Consistent-change data (CCD) table” on page 543.

### **Replicating the net change for a row to the target table**

**Target table type:** internal CCD

If changes occur frequently at a source table, you can create an internal CCD table to summarize the committed changes that occurred at the source since the last Apply cycle. Because the CD table is constantly in flux when the Capture program appends changes from the log, the local cache of source changes in the CCD acts as a more stable source for your targets.

When the original source table is updated, the Capture program reads the frequent changes in the source’s log and adds them to the source’s CD table. From that CD table, an Apply program reads the changes in the CD table and populates the internal CCD table. You can define the internal CCD table to contain only the most recent change for each row in the CD table that occurred during the last cycle. Therefore, the CCD table is static between Apply cycles (for the Apply program replicating from the CD table to the CCD table) and thus makes a more stable source for targets. By condensing changes from the source, you can improve overall replication performance by not replicating many updates for the same row to the target table.

Because the Capture program is constantly adding new changes to the CD table, a second Apply program reads changes from the internal CCD table, instead of the CD table, so that it doesn’t replicate different changes to different targets and can keep the targets in synch with one another. The second Apply program uses the original source table for full refreshes, and it uses the internal CCD table for change-capture replication.

### **Recommendations:**

- Define a subscription-set member between the source table and the internal CCD table *before* defining other subscription-set members between the source table and other target tables. That way, the Apply program will use

the internal CCD table rather than the CD table for replicating changes from the source table. If you define other subscription-set members and begin replication using those members before you define the internal CCD table for the source table, you might have to perform a full refresh for all targets of the source table.

- Combine all internal CCD tables into one subscription set to ensure that all target tables for the source database are in synch with one another.
- Even if you only want a subset of the frequently changing source columns to be applied to other targets, use the default that all registered source columns are replicated to the internal CCD. That way, you can use the internal CCD table as a source for future target tables that might need data from the other registered columns in the original source table. Only columns in the internal CCD table will be available for change-capture replication for any future target.

You use an internal CCD table as an implicit source for replication; you cannot explicitly define it as a replication source. When you add a subscription-set member, you map the original source table (not the internal CCD table) to the target table. An internal CCD table has the following attributes:

**Internal**

The CCD table acts as an alternative to the source's CD table. Information about the internal CCD table is stored in the same row as its source table in the register (IBMSNAP\_REGISTER) table; an internal CCD table does not have its own row in the register table. The Apply program automatically replicates changes from an internal CCD table, if one exists, rather than from CD tables. Only one internal CCD table can exist for each replication source.

**Local** The CCD table is in the same database as the source table.

**Noncomplete**

Because the Apply program uses the original source table for full refreshes and not the internal CCD, the CCD is noncomplete because the subsequent target will already have an initial copy of all the source rows.

**Condensed**

The internal CCD is condensed, meaning that table contains one row for every key value, so that the Apply program applies the most recent change for each row in the CCD table, instead of applying a row for every change.

**No UOW columns**

Internal CCD tables do not support additional UOW table columns. You cannot use an internal CCD table if you already defined a target CCD table that includes UOW columns.

**Important for update-anywhere:** If you define an internal CCD table, the Apply program ignores it when processing a subscription set with a replica as a target, and it applies changes to the replica from the master source's CD table.

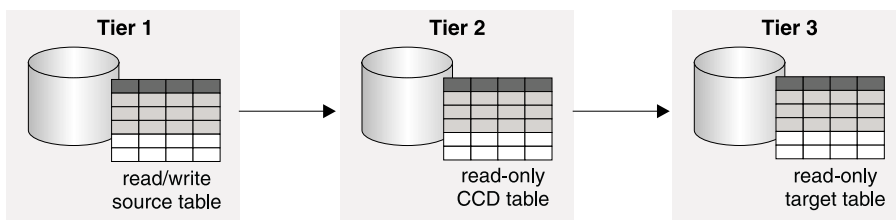
## Defining middle tiers in a multi-tier configuration

**Target table type:** CCD

The basic replication model is a two-tier model, with a single source and one or more targets; but you can set up configurations with three (or more) tiers. A multi-tier configuration has a source table and a target table, and then that target table acts as a source to other target tables.

One reason to set up a multi-tier replication environment is to provide stable sources for the third-tier targets. Because you can collect changes from tier 1 in CCD tables at tier 2, you can control how often you replicate changes to each tier and reduce the number of changes replicated to the target (tier 3). You can also avoid many of the database connections to your source system, thus moving the connection cost to the second tier.

For example, in a three-tier model, the first tier (tier 1) is the source database, the second tier (tier 2) is the target for tier 1. Tier 2 is also a source for a third tier of targets (tier 3), and can distribute changes to one or many tier-3 databases. When you have more than two tiers in your replication configuration, the middle tiers, which act as both sources and targets, are CCD tables.



*Figure 6. Three-tier replication model.* You can replicate data from a source table to a target table, and then from that table to another target table.

### Restriction:

You cannot use a non-DB2 relational table or a CCD table in a non-DB2 relational database as a middle tier in multi-tier configurations.

### Procedure:

To set up multi-tier replication, so that your target table acts as a source to subsequent targets:

1. Register the source table (tier 1) for replication. For information on how to register a table for replication, see “Registering DB2 tables as sources” on page 37.

The Capture program for this source captures changes that occur at tier 1 and stores them in tier 1’s CD table.

2. Create a subscription set between the source server and the target server (for tier 2). For information on how to create a subscription set, see “Creating subscription sets” on page 66.

The Apply program for this subscription set applies changes from tier 1 to the CCD table at tier 2.

3. Define a subscription-set member mapping the source table (tier 1) and a CCD target table (tier 2). For information on how to define subscription-set members, see “Mapping source tables and views to target tables and views within a subscription set” on page 75.

When defining the target table for this member, select for the target table to be a CCD table with the following attributes:

#### **External registered source**

You must define it as an external target table and register the table so it can act as a source for the subsequent tier. Like other registered sources, an external CCD table has its own row in the register (IBMSNAP\_REGISTER) table. External CCD tables that also act as sources can be populated only by a single source table.

You must register all external CCD tables in a subscription set using the same Capture schema.

#### **Complete**

You must use a complete CCD table because the Apply program will use this table to perform both full refresh and change-capture replication for the subsequent tier.

#### **Condensed**

Use a condensed CCD, meaning that table contains one row for every key value, to ensure that only the most recent changes are replicated to the subsequent tier. The Apply program applies the most recent change for each row in the CCD table, instead of applying a row for every change. Because condensed tables require unique key values for each row, you *must* define a unique index.

4. Because the CCD table is registered, create the Capture control tables in the middle-tier database, if they do not already exist.
5. Create a subscription set between the tier 2 server that contains registered CCD table and the subsequent target server (for tier 3). For information on how to create a subscription set, see “Creating subscription sets” on page 66.

The Apply program for this set applies changes from the CCD table to the target tables in the subsequent tier. The Apply program uses the CCD table for both full refresh and change-capture replication. Usually, you use a different Apply qualifier than the one used to populate the CCD, but you can use the same one.

6. Define a subscription-set member mapping the CCD source table (tier 2) and the subsequent target table (tier 3). For information on how to define subscription-set members, see “Mapping source tables and views to target tables and views within a subscription set” on page 75.

You can set up multiple members with target tables that subscribe to this CCD source table. If this is the final tier in your multi-tier configuration, then the target table can be any type. However, if you plan to have more than three tiers, define the tier-3 target table as specified in step 3, and repeat steps 4 through 5 to add subsequent tiers.

**Important:** If a full refresh occurs on the external CCD (the middle tier), then the Apply programs for all subsequent tiers that use that external CCD as a source will perform full refreshes. This is called a *cascade full refresh*.

## Defining read-write targets (update-anywhere)

**Target table type:** Replica

In update-anywhere replication, changes at the master source table are replicated to dependent target tables that are of type replica, and changes at the replica tables can be replicated back to the master source table. In update-anywhere replication, the master table and its replicas are read-write tables that all act as both sources and targets.

### Prerequisites:

The following conditions must exist for update-anywhere replication:

- You must use declarative referential-integrity constraints because no single application program updates both master and replica tables. Referential-integrity violations cannot be detected in application logic.
- You must include all referential constraints that exist among the master tables in the replica tables to prevent referential-integrity violations. If you omit some referential constraints, an update made to a replica table could cause an referential-integrity violation when it is replicated to the master table. The administration tools do not copy referential-constraint definitions from a source table to target tables, nor can they generate new constraints.
- To bypass referential-integrity checking during full refresh, you must use the ASNLOAD exit routine.

### Restrictions:

- You cannot use CCD tables as sources or targets in update-anywhere replication
- Columns of LOB data type cannot participate in update-anywhere replication.
- Columns of DATALINK data type cannot participate in update-anywhere replication, except when you register the source table with no conflict detection.
- Non-DB2 databases cannot have replica target-table types and, therefore, cannot participate in update-anywhere replication.

### **Procedure:**

To set up an update-anywhere configuration between a master table and one or more replica tables (where each replica table is in a separate database):

1. Because the Capture program will capture changes for each replica table, create the Capture control tables in each database that will contain a replica table, if they do not already exist.
2. Register the source table (the master table) for replication. For information on how to register a table for replication, see “Registering DB2 tables as sources” on page 37.

The Capture program for this source captures changes at the master table and stores them in the master’s CD table.

3. Create a subscription set between the master database and the target database that will contain the one or more replicas. For information on how to create a subscription set, see “Creating subscription sets” on page 66.

If all replica tables are in the same database and all master tables are in another database, you need only one subscription set. If the replica tables are in multiple databases, you need as many subscription sets as you have replica databases.

4. Define a subscription-set member for each mapping between each master table and its associated replica table. For information on how to define subscription-set members, see “Mapping source tables and views to target tables and views within a subscription set” on page 75.

In this configuration, there is only one Apply program, which typically runs at the server that contains the replica tables. The Apply program for this set pulls the changes from the master’s CD table and applies them to the replica tables. The Apply program also pushes changes from the replica table’s CD table and applies them to the master table.

**Important:** Because the master table and replica tables in update-anywhere configurations replicate data back and forth to one another, replica target tables should contain the same columns as the source table. You can create a replica target that contains a subset of the columns in the master table

only if the missing columns are defined as nullable or NOT NULL WITH DEFAULT at the master site, but you should not add new columns or rename columns at the replica.

5. Define source properties for the replica table.

When you create a subscription-set member with a replica table, DB2 replication automatically registers the replica table as a replication source. Because replica target tables act as sources, they have properties that you can set in addition to the common target table properties, which determine how the Capture program handles changes to the replica. There are two properties, however, that are inherited from the master table and cannot be changed for the replica table: the conflict-detection level and whether full refreshes are disabled. The Capture program for this source captures changes at the replica table and stores them in the replica's CD table. See "Registration options for source tables" on page 41 for a complete list of registration options, their defaults, and an explanation of when you might want to use or change the defaults.

**Important:** Even though the master and replica act as both sources and targets, full-refresh copying occurs only from the master to the replica, not from replica to master.

To prevent conflicts, you must make the target key for the replica tables the same as the master source table's primary key or unique index. Because the master table can update the replicas and the replicas can update the master, there is a potential for conflicts to occur if an update is made to a row in the master table and a different update is made to the same row in one or more replica tables between Apply cycles (so that the changes are in the master CD table and the replica CD table). A replica table inherits the level of conflict detection from the master source table or view. It is best to design your application so that a conflict can never occur when data is replicated from the master to all of the replica tables. When you registered the master source, you had three levels of conflict detection to choose from. For more information on selecting a level of conflict detection and on how to deal with conflicts that occur (if you selected either standard or enhanced conflict detection), see "Setting conflict detection (update-anywhere replication)" on page 54.

If you defined referential integrity constraints for the source table, you must define the same referential integrity constraints for the replica table to prevent integrity violations. If a referential-integrity violation occurs, the subscription cycle is automatically retried.

## Using an existing table as the target table

You can use a previously-defined DB2 table as the target table in a subscription set. That is, you can define a subscription-set member to include a target table that you defined outside of DB2 replication. Such a user-defined target table can be any of the valid target-table types for replication (user copy, point in time, base or change aggregate, CCD, or replica) as long as the

structure of the table is valid. For example, a user-defined point-in-time table must include a column of type `TIMESTAMP` called `IBMSNAP_LOGMARKER`.

**Requirements:**

- If the subscription-set member definition contains fewer columns than are in the existing target table, the target-table columns that are not involved in replication must allow nulls or be defined as `NOT NULL WITH DEFAULT`.
- There must be a unique index for point-in-time, user copy, replica, and condensed CCD tables. When you define the subscription-set member using the existing target table, you can use the existing unique index or specify a new one.

**Restrictions:**

- A subscription-set member definition cannot contain more columns than are in the existing target table.
- If you are using the Replication Center, you cannot add a column to a subscription-set member if that column does not already exist in the target table.

DB2 replication checks for inconsistencies between your existing target table and the subscription-set member definition.

**Important for multi-tier:** If you want to set up a multi-tier configuration with a source table as tier 1, a CCD table as tier 2, and an existing table as tier 3, define the CCD table to match the attributes specified for the existing target table when defining the subscription-set member between tier 1 and tier 2. Then define a subscription-set member for the existing target table in which the CCD table is the source table.

---

## Common properties for all target table types

This section discusses the common properties that you can set when creating a target table, regardless of type. You can modify properties for your target table or view based on the type of replication that you want. The following sections explain the common characteristics that you can define for how the source data maps to the target tables:

- “Source columns that you want applied to the target” on page 91
- “Source rows that you want applied to the target” on page 91
- “How source columns map to target columns” on page 92
- “Target key” on page 93
- “How the Apply program updates the target key columns with the target-key change option” on page 94



## Source columns that you want applied to the target

**Default:** all registered source columns are replicated to the target

In some replication scenarios, you might not want to replicate all columns to the target table, or the target table might not support all data types defined for the source table. You can define a column (vertical) subset that has fewer columns than your source table.

By default, your target table contains all the registered columns from the source table, except LOB and DATALINK columns. If you don't want the target table to contain all of the columns that exist in the source table, select *only* those source columns that you want to replicate to the target table. The registered columns in the source table that you do not select are still available for other subscription-set members, but are not included for the current source-to-target mapping.

You can also add calculated columns to a target table. These columns can be defined by SQL scalar functions, such as SUBSTR, or they can be derived columns, such as the division of the value of column A by the value of column B (colA/colB). These calculated columns can refer to any columns from the source table.

## Source rows that you want applied to the target

**Default:** all source rows are replicated to the target

By default, your target table contains all the rows in the source table. For some replication scenarios, you might not want to replicate all rows from the source table to the target table, or you might want to replicate source rows containing different sorts of data to different target tables. You can define a row (horizontal) subset that contains rows matching a certain condition (an SQL WHERE clause). The SQL predicate can contain ordinary or delimited identifiers. See the *DB2 SQL Reference* for more information about WHERE clauses.

### Examples:

- Assume that your target table is an operational data store for one of your company's operational divisions. You can define a WHERE clause in the subscription-set member to replicate all rows for the division (or all departments in the division) from the source table to the target table.
- Assume that you have several target tables in the same database. You can define a WHERE clause in one subscription-set member to replicate all LOB columns (plus the primary-key column) to one target table, and you can define a WHERE clause in another subscription-set member to replicate all other columns to a separate target table. Thus, your target database can

have all of the data from the source table, but denormalize the source table in the target database to adjust query performance for a data warehouse.

### **Row Predicate Restrictions:**

- Do not type WHERE in the clause; it is implied. Type WHERE in the clause only for subselect statements.
- Do not end the clause with a semicolon (;).
- If you want to use before-image columns, computed columns, or IBMSNAP columns to subset or filter your data, see “Subsetting data during subscription” on page 109.
- If your WHERE clause contains the Boolean expression OR, enclose the predicate in parentheses; for example, (COL1=X OR COL2=Y).
- If the target table is a change aggregate table and contains before-image columns, you must include the before-image columns in a GROUP BY clause.

The following examples show WHERE clauses that you can use to filter rows of the target table. These examples are very general and are designed for you to use as a model.

- WHERE clause specifying rows with specific values  
To copy only the rows that contain a specific value, such as MGR for employees that are managers, use a WHERE clause like:  
`EMPLOYEE = 'MGR'`
- WHERE clause specifying rows with a range of values  
To copy only the rows within a range, such as employee numbers between 5000 and 7000 to the target table, use a WHERE clause like:  
`EMPID BETWEEN 5000 AND 7000`

### **How source columns map to target columns**

**Default:** source column name maps to same target column name (if target table does not yet exist)

By default, the column names in the target table (if it does not yet exist) will match the column names in the source table, and the data value in a source column will be replicated to the target column with the same name. You can change the names of all columns in your target tables except the replication control columns (which begin with IBMSNAP or IBMQSQ). If the target table exists, you must map each column explicitly.

If you are mapping a DB2 table to a non-DB2 relational table with an existing nickname for the non-DB2 relational table, the data types of some columns might not be compatible. If the data types of the source columns are not compatible with the data types in the target columns, you can modify the data type at the target to make it compatible with the source:

- You can add calculated columns to adjust the data types from the source to match the required data type for the target.
- You can alter the nickname for a non-DB2 relational target table to change the data-type conversions.

**Example:** You want to replicate data from a DB2 source table with a DB2 column of data type DATE to an Oracle target table with an Oracle column of data type DATE.

*Table 4. Mapping a DB2 DATE column to an Oracle DATE column*

DB2 Column	Nickname Data Mapping	Oracle Column
A_DATE DATE	A_DATE TIMESTAMP A_DATE DATE	A_DATE DATE

The Oracle target table is created with an Oracle data type of DATE (which can contain both date and timestamp data). The initial nickname for an Oracle DATE data type in a federated database maps the DB2 data type as a TIMESTAMP. The DB2 Replication Center and the OS/400 system commands for replication alter the nickname data type to DATE, so that a DATE is replicated to Oracle and not a TIMESTAMP.

## Target key

**Default index name:** The default name comes from the target object profile for the target server, if there is one. If you have not set this profile, the default is IX + name of target table. For example, if the name of your target table is TGEMPLOYEE, the name of your target table index defaults to IXTGEMPLOYEE.

When a condensed target table is involved in change-capture replication, the Apply program requires it to have a primary key or unique index, which is called the *target key*. You can choose which columns you want to use as the unique index for your target table. The following types of target tables are condensed and require a target key:

- User copy
- Point-in-time
- Replica
- Condensed CCD

If you are creating a new target table, you can use the default index name and schema or change the defaults to match your naming conventions.

To create a unique index for a new target table, you have two options:

- Specify the columns you want as the unique index for the target table.
- Have DB2 replication select a unique index for you.

If you do not select columns for the unique index, DB2 replication checks the source table for one of the following definitions, in the following order:

1. A primary key
2. A unique constraint
3. A unique index

If DB2 replication finds one of these definitions for the source table and the associated columns are registered and part of the target table, DB2 replication uses the source table's primary key (or unique index or RRN) as the target key.

For an OS/400 source table that does not have a primary key or unique index, modify the registration for that table to use the relative record number (RRN) as a uniqueness factor. When you define the subscription-set member, specify the RRN column as the unique index for the target table. See "Using relative record numbers (RRN) instead of primary keys (OS/400)" on page 57 for details on defining an RRN for an OS/400 source table.

For target tables on OS/400 systems that use the RRN as the target key, you should run the Apply program on an OS/400 system to replicate to these target tables.

For existing target tables, you must select the unique index. You can select one of the following options:

- Use an index that already exists for the target table.  
To use an existing index, select the columns that represent the index in the Replication Center. If the Replication Center finds an exact match then it only sets a target key for the Apply program to use, otherwise it creates the unique index and sets a target key for the Apply program to use.
- Create another index for the target table.  
The unique index will be created if it does not already exist, and the target key will be set for the Apply program to use.

**Important:** If you select a key for the target table that includes columns that can be updated at the source table, you must instruct the Apply program to make special updates to the target key columns. See "How the Apply program updates the target key columns with the target-key change option" for more information.

### **How the Apply program updates the target key columns with the target-key change option**

**Restriction:** You cannot use the target-key-change option for source tables that are registered to capture updates as delete/insert pairs.

If you choose the target-key change option when you define the subscription-set member, then the Apply program makes special updates to the target key columns when the target key changes. In order for the Apply program to make these special updates, the columns that are in the source table that are part of the target-key columns for the target table must be registered with the before-image columns in the CD (or CCD) table. If you did not define the source registration to capture the before-image values of the columns that make up the target key, then you must alter your registration to include them before subscribing to a target table with a different key.

After you ensure that the before-image values of the target key columns are in the CD (or CCD) table, select the subscription-set member option for the Apply program to use the before-image values when updating target key columns.

If you do not specify for the Apply program to use the before-image values when updating target key columns, DB2 replication will not replicate data correctly when you update the columns in the source table that are part of the target key. The Apply program attempts to update the row in the target table with the new value, but it does not find the new key value in the target table to update it. The Apply program then converts the update to an INSERT and inserts the new key value in the target table. In this case, the old row with the old key value remains in the target table (and is unnecessary). When you specify that you want changes to target key columns to be processed using before-image values, the Apply program is able to find the old key value, delete it, and insert the new value.

**Related concepts:**

- Chapter 14, “Using the DB2 Replication Center” on page 243

**Related tasks:**

- Chapter 3, “Registering tables and views as replication sources” on page 37
- Chapter 6, “Subsetting data in your replication environment” on page 107
- Appendix A, “UNICODE and ASCII encoding schemes on z/OS” on page 641

**Related reference:**

- “ADDDPRSUBM: Adding a DPR subscription-set member (OS/400)” on page 376
- “ADDDPRSUB: Adding a DPR subscription set (OS/400)” on page 359
- “Consistent-change data (CCD) table” on page 543



---

## Chapter 5. Replicating special data types

When you replicate special data types, such as LOB, DATALINK, ROWID, or non-DB2 data types, you should be aware of certain conditions and restrictions. In some cases, you might have to perform additional setup steps to get DB2 replication to work with these data types. This chapter discusses these conditions and restrictions and includes the following sections:

- “General data restrictions for replication”
- “Replicating large objects” on page 98
- “Replicating DATALINK values” on page 99

---

### General data restrictions for replication

Currently, DB2 replication has specific restrictions for certain data types.

- **General data compression restrictions**

DB2 replication *cannot* replicate data that is modified using an EDITPROC or FIELDPROC clause.

- **Data encryption restrictions**

DB2 replication *cannot* replicate data that is encrypted.

- **Data type restrictions**

DB2 replication *cannot* replicate the following data types under any circumstances:

- LOB columns from non-DB2 relational sources
- Any column on which a VALIDPROC is defined

DB2 replication *can* replicate long variable graphic (LONG VARGRAPHIC) data if the source and target tables reside in DB2 for z/OS.

DB2 replication *cannot* replicate a table that contains abstract data types.

DB2 replication *can* replicate tables with spatial data type columns but *cannot* replicate the actual spatial data type columns.

User-defined data types (distinct data types in DB2 Universal Database) are converted to the base data type in the change-data (CD) table before replication.

---

## Replicating large objects

DB2 Universal Database supports large object (LOB) data types, which include: binary LOB (BLOB), character LOB (CLOB), and double-byte character LOB (DBCLOB). This section refers to all of these types as LOB data.

The Capture program reads the LOB descriptor in the log records to determine if any data in the LOB column has changed and thus should be replicated, but does not copy the LOB data to the change-data (CD) tables. When a LOB column changes, the Capture program sets an indicator in the CD table. When the Apply program reads this indicator, the Apply program then copies the entire LOB column (not just the changed portions of LOB columns) directly from the source table to the target table.

Because a LOB column can contain up to two gigabytes of data, you must ensure that you have sufficient network bandwidth for the Apply program. Likewise, your target tables must have sufficient disk space to accommodate LOB data.

### Restrictions:

- The Apply program always copies the most current version of a LOB column directly from the source table (not the CD table), even if that column is more current than other columns in the CD table. Therefore, if the LOB column in the target row changes, it is possible that this LOB column could be inconsistent with the rest of the data in that target row. To reduce this possibility of inconsistent data in the target row, ensure that the interval between the Apply cycles is as short as practical for your application.
- You can replicate 10 LOB columns or fewer per table. If you register a table with more than 10 LOB columns, the Apply program returns an error message. The Replication Center returns an error message if you attempt to register more than 10 LOB columns per table.
- You can copy LOB data only to read-only tables. Thus, you cannot replicate LOB data to replica tables.
- To copy LOB data between DB2 for OS/390 Version 6 (or later) and DB2 Universal Database (for any other operating system), you need DB2 Connect 7 or later.
- You cannot refer to LOB data using nicknames.
- Before-image values for LOB, DATALINK, or ROWID columns are not supported.
- Replication is not supported for DB2 Extenders™ for Text, Audio, Video, Image, or other extenders where additional control files associated with the extender's LOB column data are maintained outside of the database.



- DB2 replication can replicate a full LOB only; it cannot replicate parts of a LOB.
- You cannot replicate LOB columns if you use a remote journal setup in your replication environment on OS/400.

---

## Replicating DATALINK values

Accessing large files (such as multimedia data) over a remote network can be inefficient and costly. You can access and replicate unstructured files more quickly by using the DATALINK data type to represent data that is stored in external file systems.

DB2 Universal Database supports the DATALINK data type that allows the database to manage the access control, referential integrity, and recovery of these large and unstructured files. DB2 Universal Database supports DATALINK values on the following operating systems:

- AIX
- Solaris™ Operating Environments
- Windows NT
- Windows 2000
- OS/400

A DATALINK column value contains a Uniform Resource Locator (URL) that points to the location of an external file. DB2 replication uses the following components when replicating DATALINK column values and the files that they reference:

### **ASNDLCOPY exit routine**

Maps the URL on the source file system to the URL on the target file system and then connects to the appropriate file-copy daemon to replicate the referenced file.

### **Data Links Manager replication daemon (DLFM\_ASNCOPYD)**

Works with the ASNDLCOPY exit routine to copy the files that are referenced by DATALINK column values. The DLFM\_ASNCOPYD daemon is part of DB2 Data Links Manager Version 8. You can use this daemon on AIX, Solaris™ Operating Environment, and Windows operating systems.

### **ASNDLCOPYD daemon**

Works with the ASNDLCOPY exit routine to copy the files that are referenced by DATALINK column values and is shipped with DB2 for iSeries. Use the ASNDLCOPYD daemon on OS/400 and optionally on other operating systems.

When the Apply program reads data with a data type of DATALINK, the Apply program places reference data in the spill file and also places the URL of the updated file into an input file.

The Apply program then invokes the ASNDLCOPY exit routine. This ASNDLCOPY exit routine ensures that the physical file exists on the source file system, maps the URL to its corresponding file on the target file system, stores this target file location in a result file, and then connects to the appropriate file-copy daemon (DLFM\_ASCOPYD, ASNDLCOPYD, or FTP) to copy the external file from the source file system to the target file system.

**Recommendation:** Use a separate subscription set for DATALINK columns because the Apply program waits for the ASNDLCOPY routine to complete its processing before the Apply program completes replication of the subscription set. Any failures in copying the external files will cause replication of the entire subscription set to fail. If the subscription set fails, the Apply program will not deactivate the subscription set but will process the subscription set again during the next Apply cycle.

**On UNIX and Windows:** Start the Apply program with the **loadxit** parameter set to y to invoke the ASNLOAD exit routine. The ASNLOAD exit routine copies external files (to which the DATALINK values point) during a full refresh. See “Refreshing target tables using the ASNLOAD exit routine” on page 154 for more information.

**On OS/400:** Modify the ASNLOAD exit routine to call the ASNDLCP exit routine to enable the Apply program to copy external files during a full refresh. See “Refreshing target tables using the ASNLOAD exit routine” on page 154 for more information.

**Important:** Because external files can be very large, you must ensure that you have sufficient network bandwidth for both the Apply program and the file-transfer mechanism that you use to copy these files. Likewise, your target system must have sufficient disk space to accommodate these files.

**Restrictions:**

- You cannot replicate DATALINK columns between DB2 databases on OS/400 and DB2 databases on other operating systems.
- On the OS/400 operating system, there is no support for the replication of the “comment” attribute of DATALINK values.
- If you use update-anywhere replication with DATALINK columns, you must specify **None** for the conflict-detection level to turn off the conflict detection for both the DATALINK columns and the other columns in the same subscription set. DB2 replication does not check update conflicts of external files referenced by DATALINK columns.

- Before-image values for DATALINK columns are not supported.
- Target tables that are base-aggregate or change-aggregate tables cannot support DATALINK columns.
- When replicating data in consistent-change data (CCD) tables, the following restrictions apply:
  - Internal CCD tables can contain DATALINK indicators (VARCHAR type character strings that contain information about the associated URLs) but not DATALINK values. The Apply program does not invoke the ASNDLCOPY exit routine when replicating data in these table types.
  - Condensed external CCD tables can contain DATALINK columns.
  - Noncondensed CCD target tables cannot contain any DATALINK columns.

The following sections discuss the user exit routine and the file-copy daemons that the Apply programs use (depending on the operating system) to replicate both the DATALINK values and the referenced files to the target system:

- “Setting up and using the ASNDLCOPY exit routine”
- “Setting up and using DLFM\_ASNCOPYD (UNIX, Windows)” on page 103
- “Setting up and using ASNDLCOPYD (OS/400)” on page 105

## Setting up and using the ASNDLCOPY exit routine

When a subscription set is ready to be replicated, the Apply program identifies the applicable rows in the change-data (CD) table. If any DATALINK column values are found, the Apply program places the URLs of the updated files into the input file. The Apply program then invokes the ASNDLCOPY exit routine, which reads this input file and maps each DATALINK source file location to its corresponding target file location. Then, the ASNDLCOPY exit routine connects to the file-copy daemon and replicates the referenced file from the source file system to the newly mapped target file system location.

When the ASNDLCOPY routine completes, it passes a return code to the Apply program. A nonzero return code tells the Apply program that replication failed for one or more of the files; in this case, the Apply program issues a message, skips the current subscription set, and processes the next subscription set. A zero return code tells the Apply program that replication was successful.

You can use the source code for the ASNDLCOPY exit routine, and modify the sample program (which is called ASNDLCOPY.smp and is located in the \sqlib\samples\repl directory) to meet the requirements of your system. The sample program contains the following configuration files:

## ASNDLSRVMAP

Maps the source URL to the target URL.

**Example:** http://source.com/file to http://target.com/file

## ASNDLUSER

Contains the logon and address location information used when connecting to source and target file systems.

## ASNDLPARM

Contains operational parameters used to control the function of the ASNDLCOPY exit routine. These parameters include the **REPLACE\_FILE** parameter, which is used to replicate a source file to a different target file location, and the **PRESERVE\_MODTIME** parameter, which is used to preserve the last modification time of the files you are replicating. ASNDLPARM is an optional configuration file that is used only on UNIX and Windows operating systems.

You can configure your own exit routine to replicate the external files, but you must name the program ASNDLCOPY. Place the configuration files in the current execution path of the Apply program.

See the PROLOG section of the sample program in the \sqlib\samples\repl directory for information on how to set up the configuration files and to modify this exit routine.

### Procedure:

To use the ASNDLCOPY exit routine:

1. Modify the ASNDLCOPY routine, if necessary, to meet the requirements of your site.

If you turn on the trace option in the Apply program, the ASNDLCOPY routine creates two files: a log file and a trace file. The log file has the following name:

*ASNDLApplyQualSetNameSrcSrvrTgtSrvr.LOG*

where *ApplyQual* is the Apply qualifier, *SetName* is the subscription-set name, *SrcSrvr* is the source-server name, and *TgtSrvr* is the target-server name. The log file contains all messages generated by the ASNDLCOPY routine. The trace file has the following name:

*ASNDLApplyQualSetNameSrcSrvrTgtSrvr.TRC*

The trace file contains any trace information generated by the ASNDLCOPY routine.

2. Configure the ASNDLUSER, ASNDLSRVMAP, and ASNDLPARM configuration files as necessary.

**On UNIX and Windows:** If the **REPLACE\_FILE** parameter is set to YES (the default) in the ASNDLPARM file and the target file already exists in the target directory, the ASNDLCOPY exit routine replicates the source file contents to a different target system file. The ASNDLCOPY exit routine copies the content of the source file directly into a temporary file, which is given a name that is equal to the source file name with an added suffix of "new." (You can change this suffix in the ASNDLPARM file.) The Apply program then receives the URL of the original target file and the URL of the temporary file from the result file. When the Apply program propagates changes to the target table, DB2 renames the temporary file to the file name in the original target URL as the replication transaction is committed.

3. If you modified the ASNDLCOPY exit routine, compile the program and place the executable in the appropriate directory.

Because the Apply program calls the ASNDONE exit routine after subscription processing completes regardless of success or failure, you can use the routine to perform any necessary clean up if the ASNDLCOPY routine fails to replicate any external files.

### **Setting up and using DLFM\_ASNCOPYD (UNIX, Windows)**

If you installed DB2 Data Links Manager Version 8, you can use the Data Links Manager replication daemon (DLFM\_ASNCOPYD) to copy the files that are referenced by the DATALINK data type.

After the ASNDLCOPY exit routine maps the source and target URLs, this exit routine connects to a daemon to copy the files. You can configure the ASNDLUSER configuration file, specifying the address and port number needed to connect to the file-copy daemon that you want to use. You can use any FTP daemon or the DLFM\_ASNCOPYD file-copy daemon.

Both the FTP and the DLFM\_ASNCOPYD daemons copy external files from the source file system to the target file system. However, the DLFM\_ASNCOPYD file-copy daemon provides additional functionality:

- Allows retrieval of a particular version of a file that is referenced by a DATALINK column defined as RECOVERY YES.
- Allows retrieval of files referenced by DATALINK columns defined as READ PERMISSION DB depending on the access privilege of the user.
- Provides the ability to preserve the last modification time of replicated files.

### **Restrictions for DLFM\_ASNCOPYD:**

To copy your replicated files with DLFM\_ASNCOPYD, you must use DB2 Data Links Manager Version 8 with DB2 Universal Database Version 8.

You can use the DLFM\_ASNCOPYD file-copy daemon with only the following operating systems: AIX, Solaris™ Operating Environments, Windows NT, and Windows 2000.

### **Restriction for Solaris™ Operating Environments for FTP:**

If you are replicating DATALINK column values on Solaris™ Operating Environments and are using the FTP daemon to copy the files, you must use FTP daemons that support the MDTM (modtime) command. The FTP daemons that run in the source and the target file systems must support MDTM, which displays the last modification time of a given file. If you are using Version 2.6 of the Solaris™ Operating Environment, or any other version that does not include FTP support for MDTM, you need additional software such as WU-FTPD.

### **Procedure:**

To set up the DLFM\_ASNCOPYD file-copy daemon:

1. Identify the users who require a connection to this file-copy daemon.
2. Grant authority to the users to access files based on the directory where these files are located.
3. Verify that the DLFM\_ASNCOPYD daemon is enabled and that the correct port number is specified.

This port number must match the port number that is specified in the ASNDLUSER configuration file.

For more information, see *DB2 Data Links Manager Quick Beginnings* and *DB2 Data Links Manager Administration Guide and Reference*.

The Data Links File Manager archives a new version of a source file with a DATALINK column defined as RECOVERY YES each time an application links to the file through a standard SQL operation. When the Capture program captures changes to a row with a DATALINK column defined as RECOVERY YES, the Capture program records the version number of the file and places that version number in the CD table. The Apply program reads the data changes from the CD table along with the version number and passes the URLs of the new DATALINK column values and the version number to the ASNDLCOPY exit routine. When the ASNDLCOPY exit routine connects to the DLFM\_ASNCOPYD daemon, this file-copy daemon retrieves a consistent version of the external file.

Even if a more recent version of the file exists on the source system, the Data Links File Manager provides the version of the file that is consistent with the version captured in the CD table. Therefore, the target server cannot have a version that the Capture program has not yet captured in the log.

## Setting up and using ASNDLCOPYD (OS/400)

ASNDLCOPYD is the daemon that enables authorized users to retrieve files from an OS/400 source server to an OS/400 target server after the ASNDLCOPY exit routine maps the source and target URLs. After the ASNDLCOPY exit routine maps the source and target files, it connects to the ASNDLCOPYD daemon to retrieve the files. The ASNDLCOPYD file-copy daemon is similar to a FTP daemon but provides the following functions when replicating DATALINK values:

- A command for extracting file information (such as file size and last modification time)
- A command for retrieving the content of a particular file

You can configure the ASNDLCOPY exit routine to connect to the ASNDLCOPYD file-copy daemon to replicate a DATALINK column that is defined as READ PERMISSION DB.

You can find the ASNDLCOPYD sample file in library QDP4, source file QCSRC, member ASNDLCPD. The sample file builds three programs:

### ASNDLCOPYD

The main parent program and file-copy daemon.

### ASNCHILD

The program that coordinates connections from the client to the ASNDLCOPYD daemon. ASNCHILD is part of the ASNDLCOPYD daemon, which spawns a new ASNCHILD process for each request from the client.

### ASNDLCFG

A configuration program for adding and removing user IDs and for changing user ID passwords.

**Note:** If you are currently using an ASNDLCOPYD file-copy daemon under DB2 Version 7 on OS/400 or other operating system, you can continue to use this daemon with DB2 Version 8.

### Prerequisite:

You must have root (administrator) authority to run the ASNDLCOPYD daemon.

### Procedure:

To use the ASNDLCOPYD file-copy daemon:

1. Access the ASNDLCOPYD sample program in library QDP4, source file QCSRC, member ASNDLCPD.

2. Modify the sample program to meet the requirements of your site.
3. Build the program daemon.

- a. Build the base module:

```
CRTCMOD MODULE(libraryname/ASNDLCPD) SRCFILE(QDP4/QCSRC)
          DBGVIEW(*SOURCE) SYSIFCOPT(*ALL)
```

- b. Build the child program (ASNCHILD):

```
CRTPGM PGM(libraryname/ASNCHILD) MODULE(libraryname/ASNDLCPD)
```

- c. Build the parent program (ASNDLCOPYD):

```
CRTPGM PGM(libraryname/ASNDLCOPYD) MODULE(libraryname/ASNDLCPD)
```

- d. Build the configuration program (ASNDLCFG):

```
CRTPGM PGM(libraryname/ASNDLCFG) MODULE(libraryname/ASNDLCPD)
```

where *libraryname* is any existing library name. See the PROLOG section of the sample program for more information.

4. Place the executables into the QDP4 library.
5. Modify the configuration files to meet the requirements of your site.
6. Start the ASNDLCOPYD daemon with administrator authority and superuser access. Specify both the port number and the directory that contains the configuration files.

The ASNDLCOPYD file-copy daemon creates a log file for all the messages generated by the ASNDLCOPYD program. This log file has the following name: ASNDLCOPYDYYYYMMDDHHMMSS.LOG, where YYYYYMMDDHHMMSS is the time that the daemon started running.

On OS/400, DB2 replication always replicates the most recent version of an external file that is referenced by a DATALINK column value.

#### **Related tasks:**

- Chapter 10, “Operating the Apply program” on page 139



---

## Chapter 6. Subsetting data in your replication environment

Some subsetting is usually involved in replication. When you register a replication source, you choose which columns and rows you want to replicate from the source table. When you create subscription sets, you choose which of the registered columns you want to replicate to each target table.

The basic subsetting methods are described in Chapter 3, “Registering tables and views as replication sources” on page 37 and Chapter 4, “Subscribing to sources” on page 63. This chapter describes some advanced techniques that you can use to subset data. Depending on your replication requirements, you can use these techniques to subset data at the source during registration or at the target during subscription:

- If you have only one target for a source, or if all targets need exactly the same data, then it is possible to subset or manipulate at registration because you do not need to consider potentially different needs of different targets.
- If you have one source and multiple targets, and the multiple targets have different requirements regarding the data to be applied, then it may not be possible to subset at registration. In this case, you would subset data at subscription.

Do *not* use any of these techniques if you are replicating to replica target tables. The master table and replica tables in update-anywhere configurations replicate data back and forth to one another. Replica tables can have a subset of the source table columns as long as the columns that are not used are nullable. Otherwise, replica tables *must* contain the same columns as the source table so you *cannot* subset columns, add new columns, or rename columns.

This chapter contains the following sections:

- “Subsetting data during registration”
- “Subsetting data during subscription” on page 109

---

### Subsetting data during registration

You can use advanced techniques to subset your data during registration. These techniques are especially useful if you want to capture the same subset of data once and replicate that subset to many target tables. You can choose to subset data either before or after it is captured from a registered source. The techniques in this section can be used in all replication configurations except update-anywhere or peer-to-peer replication.

Subsetting data during registration can improve replication's overall performance because it reduces the amount of data that the Capture program adds to the CD table and the amount that the Apply program reads. It also reduces storage because there are less rows in the CD table.

This section discusses the following ways that you can subset data during registration:

- “Subsetting source data using views”
- “Defining triggers on CD tables to prevent specific rows from being captured (UNIX, Windows, z/OS)”

### **Subsetting source data using views**

When you register a source, you choose the columns that you want to make available for replication. The columns that you select are captured for replication. In some cases, after you register a source for change replication, you might want to register a view of the source.

For example, assume that the Human Resources department maintains a table that contains personnel data, including salary information. To maintain a backup database, the whole personnel table is registered and subscribed to at the backup site. However, if another target site wants to subscribe to the personnel table, you might want to hide the salary information from this second subscriber. The solution is to register a view over the personnel table, and allow access privileges on only the registered view for the second subscriber, so that the salary information is protected from access. A subscription can be created on this registered view.

You can also register views that include two or more source tables. For example, if you have a customer table and a branch table, the only way to adequately subset the customers to the target correctly might be by joining the two tables so that only the customers for a certain branch are replicated to a certain target. In this case, you must take care to avoid double-deletes.

### **Defining triggers on CD tables to prevent specific rows from being captured (UNIX, Windows, z/OS)**

When you register a source, the Replication Center lets you select which columns you want captured, but it does *not* let you prevent certain changes in those rows from being replicated. In some replication scenarios, you might want to prevent certain changes in rows from being captured and replicated to the target tables. For example, if you want your target tables to contain all rows and you never want any rows deleted from them, you do not want to replicate deletes from the source.

To suppress certain changes from being captured, define triggers on your CD table. These triggers specify what changes the Capture program should ignore and not add a row to the CD table. You cannot create these triggers using the

Replication Center, but you can manually create these triggers for an existing CD table (that is, after the source is registered). Any trigger failure that shows an SQLSTATE of 99999 is ignored by the Capture program and the row is not inserted into the CD table.

**Example:** Suppose that you want all source table DELETE operations to be suppressed during replication from the table SAMPLE.TABLE, where the CD table is SAMPLE.CD\_TABLE. The following trigger suppresses any rows that are DELETE operations from being inserted into the CD table:

```
CREATE TRIGGER SAMPLE.CD_TABLE_TRIGGER
NO CASCADE BEFORE INSERT ON SAMPLE.CD_TABLE
REFERENCING NEW AS CD
FOR EACH ROW MODE DB2SQL
WHEN (CD.IBMSNAP_OPERATION = 'D')
SIGNAL SQLSTATE '99999' ('CD INSERT FILTER')
```

You might want to add the create trigger statement to the SQL that was generated during registration. You must run the modified SQL to complete the registration and to create the triggers on the CD tables.

These triggers execute every time the Capture program tries to insert a row in the CD table, so you need to consider if using triggers here will give you the best performance in your replication configuration. You can increase or decrease data throughput by adding triggers to CD tables. Use triggers on the CD table to suppress a significant number of changes at the source. If you plan to capture most of the changes, but want to suppress some of them from being replicated, you might want to suppress the unwanted rows during subscription.

---

## Subsetting data during subscription

This section describes how you can use predicates to subset rows during subscription. Subsetting data during subscription can improve replication's overall performance because it reduces the amount of data that the Apply program fetches. It also reduces storage because there are less rows in the target tables.

The Apply program uses predicates to determine what data to copy during full refresh and change-capture replication. The Replication Center allows you to specify predicate values for full refresh and change-capture replication. You might want to add additional predicate information to use *only* for change-capture replication because that information is not available during full refresh. You must add this additional predicate information to the subscription set member (IBMSNAP\_SUBS\_MEMBR) table in the UOW\_CD\_PREDICATES column through SQL that you provide.

For example, suppose that you have a registered table called ALL.CUSTOMERS, and its associated CD table is called ALL.CD\_CUSTOMERS. Assume that you want the subscription target to contain only a subset of ALL.CUSTOMERS where the ACCT\_BALANCE column is greater than 50000, and you want to maintain historical data in the target table (that is, you do not want any data deleted from the target table). Using the Replication Center, you can create the subscription set member with a PREDICATES value of 'ACCT\_BALANCE > 50000'.

You cannot use the Replication Center to prevent deletes at the target table, because the information about the type of operation is stored in the CD table and is not available at the source table or view. Therefore, you must generate the additional change-capture predicate by using an SQL statement that includes the following information<sup>5</sup>:

```
UPDATE ASN.IBMSNAP_SUBS_MEMBR SET UOW_CD_PREDICATES = 'IBMSNAP_OPERATION <>"D"'
WHERE APPLY_QUAL = 'apply_qual' AND SET_NAME = 'set_name' AND
SOURCE_OWNER = 'ALL' AND SOURCE_TABLE = 'CUSTOMERS'
```

You must set up the UOW\_CD\_PREDICATES column manually for any subscription-set member predicate that references any column that is not available during full refresh, including the before-image columns in the CD table, any overhead columns from the CD table, or any column from the UOW table.

By default, the Apply program does not join the UOW table and the CD table for user-copy target tables; it fetches and applies data directly from the CD table. If the predicate has to reference the UOW table, and the target table is a user copy, you must set the value of the JOIN\_UOW\_CD column to Y in the subscription members (IBMSNAP\_SUBS\_MEMBR) table. Setting this flag ensures that the Apply program joins the UOW and CD tables.

If you want to specify predicates that exceed 1024 bytes (the capacity of the PREDICATES column of the subscription-members (IBMSNAP\_SUBS\_MEMBR) table) for a row subset, you must use a source view.

---

5. Depending on your scenario, you might need to add columns to the update statement to ensure that you update a single row in the subscription members (IBMSNAP\_SUBS\_MEMBR) table.

---

## Chapter 7. Manipulating data in your replication environment

The data in your target tables does not need to appear exactly as it does in your source tables. You can transform or enhance your source data before it is replicated to the target tables. For example, you might want to manipulate your data in the following ways: perform data cleansing, perform data aggregation, or populate columns at the target table that do not exist at the source.

This chapter describes some advanced techniques that you can use to transform your data.

You can manipulate data either before or after it's captured from a registered source. Manipulate your data at registration instead of at subscription if you want to manipulate the data *once* and replicate transformed data to *many* target tables. Manipulate your data during subscription instead of registration if you want to capture *all* of the source data and *selectively* apply transformed data to individual targets.

In some replication scenarios, you might want to manipulate the content of the source data that is stored in the CD table. A trigger, an expression through the subscription, or a source view can all be used to get the same job done. Each method has its pros and cons. A trigger might be too costly in terms of CPU used. A view lets you set up the function once rather than in multiple subscriptions.

For example, if a particular value is missing in the source table, you might *not* want the Capture program to capture null values.

You can use triggers on your CD table to specify conditions for the Capture program to enhance the data when inserting data to the CD table. In this case, you can specify that the Capture program should insert a default value in the CD table when it encounters a null value in the source. You can use the following code to create a trigger that supplies an unambiguous default if data is missing from the source table update:

```
CREATE TRIGGER ENHANCECD
NO CASCADE BEFORE INSERT ON CD_TABLE
REFERENCING NEW AS CD
FOR EACH ROW MODE DB2SQL
WHEN (CD.COL1 IS NULL)
SET CD.COL1 = 'MISSING DATA'
END
```

Instead of the trigger, you can use the COALESCE scalar function of DB2 in a registered source view or in a subscription expression. In a registered view, the coalesce function returns the first non-null value.

**Partial sample using a source view:**

```
CREATE VIEW SAMPLE.SRCVIEW (columns) AS SELECT
    ... COALESCE(A.COL1, 'MISSING DATA') ...
FROM SAMPLE.TABLE A
```

**Partial sample using an expression:**

```
COALESCE(CD.COL1, 'MISSING DATA')
```

The Apply program can manipulate data, either before or after it applies data to the target, in the following ways:

- “Enhancing data using stored procedures or SQL statements”
- “Mapping source and target columns that have different names” on page 113
- “Creating computed columns” on page 113

---

## Enhancing data using stored procedures or SQL statements

When you define subscription set information, you can define run-time processing statements using SQL statements or stored procedures that you want the Apply program to run every time it processes a *specific* set. These run-time processes enable you to manipulate the data during replication. Such statements are useful for pruning CCD tables and controlling the sequence in which subscription sets are processed. You can run the run-time processing statements at the Capture control server before a subscription set is processed, or at the target server before or after a subscription set is processed. For example, you can execute SQL statements before retrieving the data, after replicating it to the target tables, or both.

Stored procedures use the SQL CALL statement without parameters. The procedure name must be 18 characters or less in length (for OS/400, the maximum is 128). If the source or target table is in a non-DB2 relational database, the SQL statements are executed against the federated DB2 database. The SQL statements are never executed against a non-DB2 database. The run-time procedures of each type are executed together as a single transaction. You can also define acceptable SQLSTATES for each statement.

Use the ASNDONE exit routine if you want to manipulate data after *each* set completes (not after a *specific set* completes).

---

## Mapping source and target columns that have different names

When you are creating a target table using the Replication Center, you can rename columns at the target regardless of the target-table type. Also, you can change column attributes (data type, length, scale, precision, nullable) where they are compatible. You cannot use the Replication Center to rename columns of existing target tables. If the source and target columns do not match, you can either use the Replication Center to map the columns from the source to the target, or you can create a view of the target table that contains a match to the source column names.

---

## Creating computed columns

Although you cannot change the names of columns in existing target tables, you can modify the expressions of the source columns so that they map correctly to, or are compatible with, the columns in existing target tables. Using SQL expressions, you can also derive new columns from existing source columns. For aggregate target-table types, you can define new columns by using aggregate functions such as COUNT or SUM. For other types of target tables, you can define new columns using scalar functions in expressions. If the columns in source and target tables only differ by name but are otherwise compatible, you can use the Replication Center to map one column to the other.

For example, assume that you have existing source table (SRC.TABLE) and target table (TGT.TABLE):

```
CREATE TABLE SRC.TABLE (SRC_COL1 CHAR(12) NOT NULL, SRC_COL2 INTEGER,  
    SRC_COL3 DATE, SRC_COL4 TIME, SRC_COL5 VARCHAR(25))  
CREATE TABLE TGT.TABLE (TGT_COL1 CHAR(12) NOT NULL,  
    TGT_COL2 INTEGER NOT NULL, TGT_COL3 TIMESTAMP, TGT_COL4 CHAR(5))
```

Use the following steps to map the desired target table using computed columns during subscription:

1. Use the Replication Center to map SRC\_COL1 from the source table to TGT\_COL1 in the target table. Since these columns are compatible, you do not have to use an expression to map one to the other.
2. Use the computed column expression COALESCE(SRC\_COL2, 0) and map to provide TGT\_COL2. Since SRC\_COL2 is nullable and TGT\_COL2 is NOT NULL, you must perform this step to ensure that a NOT NULL value is provided for TGT\_COL2.
3. Use the computed column expression TIMESTAMP(CHAR(SRC\_COL3) CONCAT CHAR(SRC\_COL4)) and map to provide TGT\_COL3. This column expression provides data to map to the timestamp column in the target database.

4. Use computed column expression SUBSTR(SRC\_COL5, 1,5) and map to provide TGT\_COL4.



---

## Chapter 8. Customizing and running replication SQL scripts

To create control tables, register source tables, and create subscription sets, you must run SQL scripts that are generated by the Replication Center. You can run the SQL scripts using the Replication Center, the Task Center, or you can run them from a DB2 command line. If necessary, you can modify the SQL scripts to meet your needs.

You have the option in the Replication Center to run a generated SQL script immediately or to save the generated SQL script as a task or to a file and run the script at a later time. Even if you choose to run the SQL from the Replication Center, you might *also* want to save it as a task or to a file for future reference. For example, if you save the definitions of a large replication subscription set in an SQL file, you can rerun the definitions as necessary.

When editing the generated SQL scripts, be careful not to change the termination characters. Also, do not change the script separators if there are multiple scripts saved to a file.

You might want to customize the SQL scripts for your environment to perform the following tasks:

- Create multiple copies of the same replication action, customized for multiple servers.
- Set the size of the table spaces or databases of the CD tables.
- Define site-specific standards.
- Combine definitions together and run as a batch job.
- Defer the replication action until a specified time.
- Create libraries of SQL scripts for backup, site-specific customization, or to run stand-alone at distributed sites, such as for an occasionally-connected environment.
- Edit create table and index statements to represent database objects.
- For Informix and other non-DB2 relational databases, ensure that tables are created in the table spaces that you want.
- For Microsoft SQL Server, create control tables on an existing segment.
- Review and edit subscription-set member predicates as a way of defining multiple subscription sets at one time. You can use substitution variables in your predicates and resolve the variables with programming logic.

If you run the SQL scripts from a DB2 command line, you must connect to servers manually when you run the SQL script. The script is generated with CONNECT statements. Before you run the SQL script, you must edit the SQL statements to specify the user ID and password for the server. For example, look for a line that resembles the following example and add your information by typing over the placeholders (XXXX):

```
CONNECT TO srcdb USER XXXX USING XXXX ;
```

### **Procedure:**

Use one of the following methods to run the files containing SQL scripts from a DB2 command line:

- Use this command if the SQL script has a semicolon ( ; ) as a termination character:

```
db2 -tvf filename
```

- Use this command if the SQL script has some other character as the delimiter (in this example, as in heterogeneous replication, the pound sign ( # ) is the termination character):

```
db2 -td# -vf filename
```

**Recommendation:** Always read the administration log file before running any scripts.

---

## Chapter 9. Operating the Capture program

This chapter pertains to log-based capture for DB2 databases. If you are using trigger-based capture, the triggers are created at registration, and you do not perform the operations described in this chapter.

This chapter contains the following sections:

- “Default operational parameters for the Capture program”
- “Changing operational parameters for the Capture program” on page 119
- “Starting the Capture program (UNIX, Windows, z/OS)” on page 121
- “Starting the Capture program (OS/400)” on page 132
- “Altering the behavior of a running Capture program” on page 132
- “Changing the operating parameters in the Capture parameters table” on page 134
- “Stopping the Capture program” on page 135
- “Suspending Capture (UNIX, Windows, z/OS)” on page 136
- “Resuming Capture (UNIX, Windows, z/OS)” on page 136
- “Reinitializing Capture” on page 137

**Important:** The Capture program does not capture any changes made by DB2 utilities, because the utilities do not log changes in a way that is visible to the Capture program.

---

### Default operational parameters for the Capture program

Capture has several parameters for which there are default values. The default values that are shipped with the product are shown in Table 5 and Table 6 on page 118. The default values for most operational parameters are shipped and are stored in the Capture parameters (IBMSNAP\_CAPPARMS) table. Use these defaults in your replication environment and change them as necessary using one of the methods described in “Changing operational parameters for the Capture program” on page 119.

*Table 5. Default settings for Capture operational parameters (UNIX, Windows, z/OS)*

Operational parameter	Default value	Column name in IBMSNAP_CAPPARMS table
capture_server	DB2DBDFT <sup>1</sup>	not applicable
capture_schema	ASN <sup>2</sup>	not applicable
retention_limit	10080 minutes	RETENTION_LIMIT

Table 5. Default settings for Capture operational parameters (UNIX, Windows, z/OS) (continued)

Operational parameter	Default value	Column name in IBMSNAP_CAPPARMS table
lag_limit	10080 minutes	LAG_LIMIT
commit_interval	30 seconds	COMMIT_INTERVAL
prune_interval	300 seconds	PRUNE_INTERVAL
trace_limit	10080 minutes	TRACE_LIMIT
monitor_limit	10080 minutes	MONITOR_LIMIT
monitor_interval	300 seconds	MONITOR_INTERVAL
memory_limit	32 MB	MEMORY_LIMIT
autoprune	y <sup>3</sup>	AUTOPRUNE
term	y <sup>3</sup>	TERM
autostop	n <sup>4</sup>	AUTOSTOP
logreuse	n <sup>4</sup>	LOGREUSE
logstdout	n <sup>4</sup>	LOGSTDOUT
sleep_interval	5 seconds	SLEEP
capture_path	Directory where Capture was started <sup>5</sup>	CAPTURE_PATH
startmode	warm <sup>6</sup>	STARTMODE

**Notes:**

1. The Capture control server is the value of the DB2DBDFT environment variable, if specified.
2. You cannot change the default for the Capture schema. To use another Capture schema, use the **capture\_schema** start-up parameter.
3. Yes
4. No
5. If Capture starts as a Windows service, its capture path is sqllib\bin.
6. The Capture program warm starts. It switches to cold start only if this is the first time that the program is starting.

For more information about these operational parameters and their defaults, see “Starting the Capture program (UNIX, Windows, z/OS)” on page 121.

Table 6. Default settings for Capture operational parameters (OS/400)

Operational parameter	Default value	Column name in IBMSNAP_CAPPARMS table
CAPCTLLIB	ASN <sup>1</sup>	not applicable

Table 6. Default settings for Capture operational parameters (OS/400) (continued)

Operational parameter	Default value	Column name in IBMSNAP_CAPPARMS table
JOBD	*LIBL/QZSNDPR	not applicable
JRN	*ALL	not applicable
RETAIN	10080 minutes	RETENTION_LIMIT
LAG	10080 minutes	LAG_LIMIT
FRCFRQ	30 seconds	COMMIT_INTERVAL
CLNUPITV	*IMMED <sup>2</sup>	not applicable
CLNUPITV	86400 seconds <sup>2</sup>	PRUNE_INTERVAL
CLNUPITV	*IMMED <sup>2</sup>	not applicable
TRCLMT	10080 minutes	TRACE_LIMIT
MONLMT	10080 minutes	MONITOR_LIMIT
MONITV	300 seconds	MONITOR_INTERVAL
MEMLMT	32 MB	MEMORY_LIMIT
WAIT	120 seconds	not applicable
RESTART	*YES <sup>3</sup>	not applicable

**Notes:**

1. You cannot change the default for the Capture schema. To use another Capture schema, specify the CAPCTLLIB parameter when you start the Capture program. The default values for most other operational parameters are shipped and are stored in the Capture parameters (IBMSNAP\_CAPPARMS) table.
2. The CLNUPITV has two sub-parameters. By default, the Capture program prunes soon after it starts running and again after every prune interval is reached (which, by default, is every 24 hours).
3. By default, the Capture program warm starts.

For more information about these operational parameters and their defaults, see Chapter 18, “System commands for replication (OS/400)” on page 349.

## Changing operational parameters for the Capture program

You can change the default values for the operational parameters to values that you typically use in your environment. You can override these default values when you start the Capture program or modify them while the Capture program is running.

### Setting new default values in the IBMSNAP\_CAPPARMS table

The Capture parameters (IBMSNAP\_CAPPARMS) table contains parameters that you can modify to control the operation of the

Capture program. The schema name of the table is the Capture schema. After the table is created, it contains the default values that are shipped for the Capture program. If the column value in the CAPPARMS table is not set, the hard-coded default value shown in Table 5 on page 117 and Table 6 on page 118 are used. For more information about how to modify the values in this table, see “Changing the operating parameters in the Capture parameters table” on page 134.

### **Specifying values for parameters when you start the Capture program**

You can specify values for the Capture program when you start it. The values that you set during startup control the behavior of Capture for the current session, they override the default operational parameter values and any values that might exist in the Capture parameters table. They do not update the values in the Capture parameters table. If you do not modify the Capture parameters table before you start the Capture program, and you do not specify any parameters when you start the Capture program, default values are used for the operational parameters.

### **Changing parameter values while the Capture program is running**

While Capture is running, you can change its operational parameters temporarily. The Capture program will use the new values until you change the values again, or until you stop and restart the Capture program. You can change the Capture parameters as often as you like during the session. See “Altering the behavior of a running Capture program” on page 132 for details.

**Example (UNIX, Windows):** Assume that you do not want to use the default settings for the Capture commit interval for Capture schema ASNPROD.

1. Update the Capture parameters table for the ASNPROD Capture schema. Set the commit interval to 60 seconds; therefore, when you start the Capture program in the future, the commit interval will default to 60 seconds.  

```
update asnprod.ibmsnap_capparms set commit_interval=60;
```
2. Eventually you might want to do some performance tuning so you decide to try starting Capture using a lower commit interval. Instead of changing the value in the Capture parameters table, you simply start the Capture program with the commit interval parameter set to 20 seconds. While the Capture program runs using a 20-second commit interval, you monitor its performance.  

```
asncap capture_server=srddb1 capture_schema=asnprod commit_interval=20
```
3. You decide that you want to try an even lower commit interval. Instead of stopping the Capture program, you submit a change parameters request that sets the commit interval to 15 seconds. The Capture program continues to run, only now it commits data every 15 seconds.

```
asnccmd capture_server=srcdb1 capture_schema=asnprod chgparms  
commit_interval=15
```

**Important:** The parameter that you are changing must immediately follow the **chgparms** command.

4. You can continue monitoring the performance and changing the commit interval parameter without stopping the Capture program. Eventually, when you find the commit interval that meets your needs, you can update the Capture parameters tables (as described in step 1 on page 120) so that the next time you start the Capture program it uses the new value as the default commit interval.

**Example (OS/400):** Assume that you do not want to use the default settings for the Capture commit interval for Capture schema ASNPROD.

1. Update the Capture parameters table for the ASNPROD Capture schema. Set the commit interval to 90 seconds; therefore, when you start the Capture program in the future the commit interval will default to 90 seconds.

```
CHGDPRCAP CAPCTLLIB(ASNPROD) FRCFRQ(90)
```

2. Eventually you might want to do some performance tuning so you decide to try starting Capture using a lower commit interval. Instead of changing the value in the Capture parameters table, you simply start the Capture program with the commit interval parameter set to 45 seconds. As the Capture program runs using a 45-second commit interval, you monitor its performance.

```
STRDPRCAP CAPCTLLIB(ASNPROD) FRCFRQ(45)
```

3. You decide that you want to try an even lower commit interval. Instead of stopping the Capture program, you submit a change parameters request that sets the commit interval to 30 seconds. The Capture program continues to run, only now it commits data every 30 seconds. (Note: On OS/400, you can't set the commit interval to less than 30 seconds.)

```
OVRDPRCAP CAPCTLLIB(ASNPROD) FRCFRQ(30)
```

4. Eventually, when you find the commit interval that meets your needs, you can update the Capture parameters tables (as described in step 1) so that the next time you start the Capture program it will use the new value as the default commit interval.

---

## Starting the Capture program (UNIX, Windows, z/OS)

Start the Capture program to begin capturing data from the log. The Capture program captures data from DB2 databases only. If you are using trigger-based capture to capture changes from a non-DB2 relational source, the triggers are created at registration and you do not need to start the Capture program.

**Important:** The Capture program does not capture any changes made by DB2 utilities, because the utilities do not log changes in a way that is visible to the Capture program.

After you start the Capture program, the Capture program might not start capturing data right away. It will start capturing data only after the Apply program signals the Capture program that it has refreshed a target table fully. Then the Capture program starts capturing changes from the log for a given source table.

**Tip:** Look in the Capture log file (*db2instance.capture\_server.capture\_schema.CAP.log* on UNIX and WINDOWS; *capture\_server.capture\_schema.CAP.log* on z/OS) for a message that indicates that change capture has begun. For example:

ASN0104I Change capture has been started for the source table "REGRESS.TABLE1" for changes found in the log beginning with log sequence number "0000:0275:6048".

### **Prerequisites:**

Before you start the Capture program, ensure that the following prerequisites are met:

- Connections are configured to the source server and the Capture control server.
- You have the proper authorization.
- The control tables are created for the appropriate Capture schema, and registrations are defined.
- The replication programs are configured.

### **Procedure:**

Use one of the following methods to start the Capture program on DB2 for UNIX, Windows, and z/OS:

#### **Replication Center**

Use the Start Capture window to run the Capture program identified by a Capture schema on a selected Capture control server that is in the Replication Center object tree. See the Replication Center help for details.

#### **asncap system command**

See “asncap: Starting Capture (UNIX, Windows, z/OS)” on page 316 for command syntax and parameter descriptions.

#### **MVS console or TSO (z/OS)**

See Chapter 19, “Operating the replication programs (z/OS)” on page 453 for details.



## Windows Services (Windows)

See Chapter 20, “Using the Windows Service Control Manager to issue system commands (Windows)” on page 459 for details.

Regardless of which procedure you use to start the Capture program, you can select start-up parameters. The following sections discuss the start-up parameters and recommend when to choose one value over another for each parameter.

- “autoprune(UNIX, Windows, z/OS)”
- “autostop (UNIX, Windows, z/OS)” on page 124
- “capture\_path (UNIX, Windows, z/OS)” on page 124
- “capture\_schema (UNIX, Windows, z/OS)” on page 125
- “capture\_server (UNIX, Windows, z/OS)” on page 126
- “commit\_interval (UNIX, Windows, z/OS)” on page 126
- “lag\_limit (UNIX, Windows, z/OS)” on page 126
- “logreuse (UNIX, Windows, z/OS)” on page 126
- “logstdout (UNIX, Windows, z/OS)” on page 127
- “memory\_limit (UNIX, Windows, z/OS)” on page 127
- “monitor\_interval (UNIX, Windows, z/OS)” on page 128
- “monitor\_limit (UNIX, Windows, z/OS)” on page 128
- “prune\_interval (UNIX, Windows, z/OS)” on page 128
- “retention\_limit (UNIX, Windows, z/OS)” on page 129
- “sleep\_interval (UNIX, Windows, z/OS)” on page 130
- “startmode (UNIX, Windows, z/OS)” on page 130
- “term (UNIX, Windows, z/OS)” on page 131
- “trace\_limit (UNIX, Windows, z/OS)” on page 131

## autoprune(UNIX, Windows, z/OS)

**Default:** autoprune=y

The **autoprune** parameter specifies whether or not the Capture program automatically prunes some of its control tables. By default, with **autoprune=y**, the Capture program automatically prunes the rows in the CD and UOW tables as well as Capture trace (IBMSNAP\_CAPTRACE), Capture monitor (IBMSNAP\_CAPMON), and signal (IBMSNAP\_SIGNAL) tables. If you set **autoprune=n**, you must use the prune command to prune these tables.

If you start Capture with autopruning on, set the prune interval to optimize the pruning frequency for your replication environment. See “prune\_interval (UNIX, Windows, z/OS)” on page 128 for details.

The Capture program uses the following parameters to determine which rows are old enough to prune:

- “retention\_limit (UNIX, Windows, z/OS)” on page 129 for CD, UOW, and signal tables
- “monitor\_limit (UNIX, Windows, z/OS)” on page 128 for monitor tables
- “trace\_limit (UNIX, Windows, z/OS)” on page 131 for the Capture trace table

For more information about pruning your tables, see “Pruning your control tables” on page 233.

### **autostop (UNIX, Windows, z/OS)**

**Default:** **autostop=n**

The **autostop** parameter controls whether the Capture program stays up or terminates after it reaches the end of the log.

By default (**autostop=n**) the Capture program does not terminate after retrieving the transactions.

Use the **autostop=y** option if you are replicating in a mobile or an occasionally connected environment. Autostop ensures that the Capture program retrieves all eligible transactions and stops when it reaches the end of the log. You need to start Capture again to retrieve more transactions. You might want to use the **autostop=y** option in a test environment, too.

**Recommendation:** In most cases you should not use **autostop=y** because it adds a lot of overhead to the administration of replication (for example, you need to keep restarting the Capture program).

### **capture\_path (UNIX, Windows, z/OS)**

The Capture path is the directory where the Capture program stores its work files and log file. By default, the Capture path is the directory where you start the program. If you start the Capture program as a Windows service, by default the Capture program starts in the \sqllib\bin directory. On the z/OS platform, because the Capture program is a POSIX application, the default Capture path depends on how you start the program:

- If you start the Capture program from a USS command line prompt, the Capture path is the directory where you started the program.
- If you start the Capture program using a started task or through JCL, the default Capture path is the home directory of the user ID associated with the started task or job.

You can change the Capture path to specify where you want the Capture program to store its files. You can specify a path name, for example:

/home/db2inst/capture\_files. On z/OS platforms, you can specify either a path name or a High Level Qualifier(HLQ), such as //CAPV8. When you use a HLQ, sequential files are created that conform to the file naming conventions for z/OS sequential dataset file names. The sequential datasets are relative to the user ID that is running the program. Otherwise these file names are similar to those stored in an explicitly named directory path, with the HLQ concatenated as the first part of the file name. For example, *sysadm.CAPV8.filename*.

## **capture\_schema (UNIX, Windows, z/OS)**

**Default:** `capture_schema=ASN`

The **capture\_schema** parameter identifies which Capture program you want to start. By default, the Capture schema is ASN.

If you already set up another schema, you can start the Capture program by specifying that schema using the **capture\_schema** parameter. See “Creating multiple sets of Capture control tables” on page 25 for instructions.

You might use multiple Capture schemas in the following situations:

### **Achieving application independence**

Create multiple Capture schemas so that you can have one Capture program for application A and another Capture program for application B. Each Capture program uses its own control tables. If one of the Capture programs is down, only one application is affected. The other application is not affected because it is being serviced by another Capture program.

### **Meeting different applications’ requirements**

Create multiple Capture schemas if you have different applications that use the same source tables but have different data requirements. For example, a payroll application needs sensitive employee data while an internal employee registry does not. You can register the confidential information in one Capture schema, but not in the other Capture schema. Similarly, you can register a table more than once if some applications need the Capture program to behave differently. For example, perhaps some applications require that the Capture program saves updates as delete and insert pairs.

### **Isolating problems with registrations**

If you have a problem with one registration, you can create another Capture schema and move the working registrations to it. That way you can debug the problem registration in the original schema and run the unaffected registrations using the other schema.

## **capture\_server (UNIX, Windows, z/OS)**

**Default (UNIX, Windows):** `capture_server`= value of DB2DBDFT environment variable, if it is set

**Default (z/OS):** `capture_server`= None

The **capture\_server** parameter specifies the Capture control server. The capture control tables (such as the register table) are located at the capture control server and they contain the registration information for the source tables. Because the Capture program reads the DB2 log, the Capture program must run at the same server as the source database.

## **commit\_interval (UNIX, Windows, z/OS)**

**Default:** `commit_interval`=30

The **commit\_interval** parameter specifies how often, in seconds, the Capture program commits data to the Capture control tables, including the UOW and CD tables. By default, the Capture program waits 30 seconds before committing data to the CD and UOW tables. Locks are held on the tables updated within the commit interval. Higher values for the **commit\_interval** parameter reduce CPU usage for the Capture program but also might increase the latency for frequently running subscription sets because the Apply program can fetch only committed data.

## **lag\_limit (UNIX, Windows, z/OS)**

**Default:** `lag_limit`=10 080

The **lag\_limit** parameter represents the number of minutes that the Capture program can lag in processing records from the DB2 log.

By default, if log records are older than 10 080 minutes (seven days), the Capture program will not start unless you specify a value for the **startmode** parameter that allows the Capture program to switch to a cold start.

If the Capture program will not start because the lag limit is reached, you should determine why the Capture program is behind in reading the log. If you are in a test environment, where you have no practical use for the lag limit parameter, you might want to set the lag limit higher and try starting the Capture program again. Alternatively, if you have very little data in the source table in your test environment, you might want to cold start the Capture program and fully refresh the data in all the target tables.

## **logreuse (UNIX, Windows, z/OS)**

**Default:** `logreuse`=n

The Capture program stores operational information in a log file.

On UNIX and Windows platforms, the name of the log file is *db2instance.capture\_server.capture\_schema.CAP.log*. For example, *DB2INST.SRCDB1.ASN.CAP.log*.

On the z/OS platform, the file name is similar except that it does not contain a DB2 instance name. For example, *SRCDB1.ASN.CAP.log*. This file is stored in the directory that is specified by the **capture\_path** parameter. If the **capture\_path** parameter is specified as a High Level Qualifier (HLQ), the file naming conventions of z/OS sequential dataset files apply; therefore, the **capture\_schema** name that is used to build the log file name is truncated to the first 8 characters of the name.

By default (**logreuse=n**), the Capture program appends messages to the log file, even after the Capture program is restarted. Keep the default if you want the history of the messages. In the following situations you might want the Capture program to delete the log and re-create it when it restarts (**logreuse=y**):

- The log is getting large and you want to clean out the log.
- You don't need the history that is stored in the log.
- You want to save space.

## **logstdout (UNIX, Windows, z/OS)**

**Default:** **logstdout=n**

The **logstdout** parameter is available only if you use the **asncap** command, it is not available in the Replication Center.

By default, the Capture program sends some warning and informational messages only to the log file. You might choose to send such messages to standard output (**logstdout=y**) if you are troubleshooting or if you are monitoring how your Capture program is operating in a test environment.

## **memory\_limit (UNIX, Windows, z/OS)**

**Default:** **memory\_limit=32**

The **memory\_limit** parameter specifies the amount of memory, in megabytes, that the Capture program can use.

By default, the Capture program uses 32 megabytes of memory to store transaction information before it spills to a file located in the **capture\_path** directory. You can modify the memory limit based on your performance needs. Setting the memory limit higher can improve the performance of Capture but decreases the memory available for other uses on your system. Setting the memory limit lower frees memory for other uses. If you set the memory limit too low and the Capture program spills to a file, you will use more space on your system and the I/O will slow down your system.

You can monitor the memory limit by using the Replication Alert Monitor. You can also use the data in the CAPMON table to determine the number of source system transactions spilled to disk due to memory restrictions. Sum the values in the TRANS\_SPILLED column of the CAPMON table.

### **monitor\_interval (UNIX, Windows, z/OS)**

**Default:** monitor\_interval=300

The **monitor\_interval** parameter specifies how often the Capture program writes information to the Capture monitor (IBMSNAP\_CAPMON) table.

By default, the Capture program inserts rows into the Capture monitor table every 300 seconds (5 minutes). This operational parameter works in conjunction with the commit interval. If you are interested in monitoring data at a granular level, use a monitor interval that is closer to the commit interval.

### **monitor\_limit (UNIX, Windows, z/OS)**

**Default:** monitor\_limit=10080

The **monitor\_limit** parameter specifies how old the rows must be in the monitor table before they can be pruned.

By default, rows in the Capture monitor (IBMSNAP\_CAPMON) table that are older than 10 080 minutes (seven days) are pruned. The IBMSNAP\_CAPMON table contains operational statistics for the Capture program. Use the default monitor limit if you need less than one week of statistics. If you monitor the statistics frequently, you probably do not need to keep one week of statistics and can set a lower monitor limit so that the Capture monitor table is pruned more frequently and older statistics are removed. If you want to use the statistics for historical analysis and you need more than one week of statistics, increase the monitor limit.

### **prune\_interval (UNIX, Windows, z/OS)**

**Default:** prune\_interval=300

The **prune\_interval** parameter specifies how often the Capture program tries to prune old rows from some of its control tables. This parameter is valid *only* if **autoprun**=y.

By default, the Capture program prunes the CD and UOW tables every 300 seconds (five minutes). If the tables are not pruned often enough, the table space that they are in can run out of space, which forces the Capture program to stop. If they are pruned too often or during peak times, the pruning can interfere with application programs running on the same system. You can set the optimal pruning frequency for your replication environment. Performance will generally be best when the tables are kept small.

Before you lower the prune interval, ensure that data is being applied frequently so that pruning can occur. If the Apply program is not applying data frequently, it is useless to set the prune interval lower because the Apply program must replicate the data to all targets before the CD and UOW tables can be pruned.

The prune interval determines how often the Capture program *tries* to prune the tables. It works in conjunction with the following parameters, which determine *when* data is old enough to prune: **trace\_limit**, **monitor\_limit**, **retention\_limit**. For example, if the **prune\_interval** is 300 seconds and the **trace\_limit** is 10080 seconds, the Capture program will try to prune every 300 seconds. If it finds any rows in the trace table that are older than 10080 minutes (7 days), it will prune them.

For more information about pruning your tables, see “Pruning your control tables” on page 233.

### **retention\_limit (UNIX, Windows, z/OS)**

**Default:** **retention\_limit=10 080**

The **retention\_limit** parameter determines how long old data remains in the CD, UOW, and signal (IBMSNAP\_SIGNAL) tables before becoming eligible for retention limit pruning.

If the normal pruning process is inhibited due to deactivated or infrequently run subscription sets, data remains in the CD and UOW tables for long periods of time. If this data becomes older than the current DB2 timestamp minus the retention limit value, the retention limit pruning process deletes this data from the tables. If you run your subscription sets very infrequently or stop your Apply programs, your CD and UOW tables can grow very large and become eligible for retention limit pruning.

Your target tables must be refreshed to synchronize them with the source if *any* of the rows that are pruned are candidates for replication but for some reason they *were not yet applied* to the target table. You can avoid a full refresh from happening by using higher retention limits; however, your CD and UOW tables will grow and use space on your system.

If you are doing update-anywhere replication, retention limit pruning ensures that rejected transactions are deleted. Rejected transactions result if you use conflict detection with replica target tables and conflicting transactions are detected. The rows in the CD and UOW tables that pertain to those rejected transactions are not replicated and they are pruned when the retention limit is reached. A full refresh is not required if *all* the old rows that were deleted pertained to rejected transactions.

Retention pruning also ensures that signal information that is no longer required is deleted from the signal (IBMSNAP\_SIGNAL) table.

For details about pruning your control tables, see “Pruning your control tables” on page 233.

### **sleep\_interval (UNIX, Windows, z/OS)**

**Default:** `sleep_interval=5`

The sleep interval is the number of seconds that the Capture program waits before it reads the log again after it reaches the end of the log and the buffer is empty. For data sharing on the z/OS platform, the sleep interval represents the number of seconds that the Capture program sleeps after the buffer returns less than half full.

By default, the Capture program sleeps 5 seconds. Change the sleep interval if you want to reduce the overhead of the Capture program reading the log. A smaller sleep interval means there is less chance of delay. A larger sleep interval gives you potential CPU savings in a sparsely updated system.

### **startmode (UNIX, Windows, z/OS)**

**Default:** `startmode=warmnsi`

You can start Capture using one of the following start modes:

#### **Warmnsi (warm start, switch initially to cold start)**

The Capture program warm starts; except if this is the first time you're starting the Capture program then it switches to cold start. Use this start mode if you want to ensure that cold starts only happen when you start the Capture program initially.

#### **Warmns (warm start, never switch to cold start)**

The Capture program warm starts. If it can't warm start, it does not switch to cold start. When you use **warmns** in your day-to-day replication environment, you have an opportunity to repair any problems (such as unavailable databases or table spaces) that are preventing a warm start from occurring. Use this start mode to prevent a cold start from occurring unexpectedly. When the Capture program warm starts, it resumes processing where it ended. If errors occur after the Capture program started, the Capture program terminates and leaves all tables intact.

**Tip:** You cannot use **warmns** to start the Capture program for the first time because there is no warm start information when you initially start the Capture program. Use the **cold** startmode the first time you start the Capture program, then use the **warmns** startmode. If you do not want to switch startmodes, you can use **warmnsi** instead.



**Warmsa (warm start, always switch to cold as necessary)**

If warm start information is available, the Capture program resumes processing where it ended in its previous run. If the Capture program cannot warm start, it switches to a cold start. Usually you do not want to switch to a cold start because that requires all your target tables to be refreshed.

**Cold** During cold start, the Capture program deletes all rows in its CD tables and UOW table during initialization. All subscription sets to these replication sources are fully refreshed during the next Apply processing cycle (that is, all data is copied from the source tables to the target tables). If the Capture program tries to cold start but you disabled full refresh, the Capture program will start, but the Apply program will fail and will issue an error message.

You rarely want to explicitly request that the Capture program performs a cold start. Cold start is necessary only the first time the Capture program starts, and **warmsi** is the recommended start mode.

**Important:** Do *not* cold start the Capture program if you want to maintain accurate histories of change data. A gap might occur if the Apply program cannot replicate changes before the Capture program shuts down. Also, because you want to avoid cold starts, do *not* put cold start as the default for STARTMODE in the Capture parameters (IBMSNAP\_CAPPARMS) table.

**term (UNIX, Windows, z/OS)**

**Default:** term=y

The **term** parameter determines how the status of DB2 affects the operation of the Capture program.

By default, the Capture program terminates if DB2 terminates.

Use **term=n** if you want the Capture program to wait for DB2 to start if DB2 is not active. If DB2 quiesces, Capture does not terminate; it remains active but it does not use the database.

**trace\_limit (UNIX, Windows, z/OS)**

**Default:** trace\_limit=10 080

The **trace\_limit** specifies how old the rows must be in the Capture trace (IBMSNAP\_CAPTRACE) table before they are pruned.

When Capture prunes, by default, the rows in the Capture trace (IBMSNAP\_CAPTRACE) table are eligible to be pruned every 10 080 minutes (seven days). The CAPTRACE table contains the audit trail information for the Capture program. Everything that Capture does is recorded in this table;

therefore this table can grow very quickly if the Capture program is very active. Modify the trace limit depending on your need for audit information.

---

## Starting the Capture program (OS/400)

Start the Capture program to begin capturing data from the journal.

After you start the Capture program, the Capture program might not start capturing data right away. It will start capturing data only after the Apply program signals the Capture program to start capturing changes from the log for a given source table.

### Prerequisites:

Before you start the Capture program, follow the instructions in Chapter 2, “Setting up for replication” on page 15 to ensure that the following prerequisites are met:

- You have the proper authorization.
- The control tables are created for the appropriate Capture schema, and registrations are defined.
- The replication programs are configured if the Capture program is reading a remote journal.

### Procedure:

Use one of the following methods to start the Capture program on OS/400:

#### Replication Center

Use the Start Capture window to run the Capture program identified by a Capture schema on a selected Capture control server that is in the Replication Center object tree. See the Replication Center help for details.

#### STRDPRCAP system command (OS/400)

See “STRDPRCAP: Starting Capture (OS/400)” on page 436 for command syntax and parameter descriptions.

---

## Altering the behavior of a running Capture program

While the Capture program is running, you can alter its behavior by overriding the values of one or more operational parameters. The changes are not written to the Capture parameters (IBMSNAP\_CAPPARMS) table. The Capture program uses the new values until you stop the Capture program or until you supply new values.

On UNIX, Windows, and z/OS, you can change the following Capture parameters while the Capture program is running:

- Autoprune
- Autostop
- Commit\_interval
- Lag\_limit
- Logreuse
- Logstdout
- Memory\_limit
- Monitor\_interval
- Monitor\_limit
- Prune\_interval
- Retention\_limit
- Sleep\_interval
- Term
- Trace\_limit

On OS/400, you can override the values for the following operational parameters for a given Capture schema:

- CLNUPITV
- FRCFRQ
- MEMLMT
- MONLMT
- MONITV
- PRUNE
- RETAIN
- TRCLMT

When you change the values, the effects might not be immediate for all parameters.

**Prerequisites:**

The Capture program with the specific Capture schema must be started.

**Procedure:**

Use one of the following methods to see the current values for the parameters and to change them for the current session:

### **Replication Center**

In the Replication Center, use the Change Operational Parameters window while the Capture program is running. See the Replication Center help for details.

### **chgparms system command (UNIX, Windows, z/OS)**

See “asnccmd: Operating Capture (UNIX, Windows, z/OS)” on page 322.

### **OVRDPRCAPA system command (OS/400)**

See “OVRDPRCAPA: Overriding DPR capture attributes (OS/400)” on page 414.

---

## **Changing the operating parameters in the Capture parameters table**

The Capture parameters (IBMSNAP\_CAPPARMS) table contains the operational parameters for the Capture program. When you start the Capture program, it uses values from this table for its default operational behavior, unless you provide new values using the start-up parameters.

Only one row is allowed in the Capture parameters table. If you want to change one or more of the default values, you can update columns instead of inserting rows. If you delete the row, the Capture program will still start using the shipped defaults, unless those defaults are overridden by the start-up parameters.

The Capture program reads this table only during startup; therefore, you should stop and start the Capture program if you want the Capture program to run with the new settings. Changing the Capture parameters table while the Capture program is running and reinitializing the Capture program will not change the operation of the Capture program. See Chapter 23, “Table structures” on page 471 for descriptions of the columns in this table.

### **Procedure:**

Use one of the following methods to change the global operating parameters that are used by the Capture program and are stored in the Capture parameters (IBMSNAP\_CAPPARMS) table:

### **Replication Center**

In the Replication Center, use the Manage Capture Parameters window to view or change any of the values in the Capture parameters table. See the Replication Center help for details.

### **CHGDPRCAPA system command (OS/400)**

See “CHGDPRCAPA: Changing DPR Capture attributes (OS/400)” on page 391.

The parameter changes take effect only after you stop and start the Capture program.

---

## Stopping the Capture program

You can stop the Capture program for a particular Capture schema. When you stop the Capture program, it no longer captures data from the source.

**OS/400:** If you choose to reorganize the UOW table and all the CD tables that were open at the time that the Capture program stopped, the Capture program needs time to shut down (it does not shut down immediately).

### Prerequisites:

The Capture program with the specific Capture schema must be started.

### Procedure:

Use one of the following methods to stop the Capture program for the specific Capture schema:

#### Replication Center

In the Replication Center, use the Stop Capture window to stop the running Capture program for the selected Capture schema. See the Replication Center help for details.

#### **asncmd stop system command (UNIX, Windows, z/OS)**

See “asncmd: Operating Capture (UNIX, Windows, z/OS)” on page 322.

#### **ENDDPRCAP system command (OS/400)**

See “ENDDPRCAP: Stopping Capture (OS/400)” on page 400.

If you stop or suspend the Capture program during pruning, pruning is also suspended. When you resume or restart the Capture program, pruning resumes based on the **autoprune** parameter.

You do not need to stop the Capture program to drop a registration. Always deactivate the registration before you drop it. For details, see “Stop capturing changes for registered objects” on page 188.

---

## Suspending Capture (UNIX, Windows, z/OS)

Suspend the Capture program to relinquish operating system resources to operational transactions during peak periods without destroying the Capture program environment. Suspend the Capture program instead of stopping it if you do not want the Capture program to shut down after it finishes work in progress. When you tell the Capture to resume, you do not require the overhead of Capture starting again.

**Important:** Do not suspend the Capture program before you remove a replication source. Instead, deactivate then remove the replication source.

### Prerequisites:

The Capture program with the specific Capture schema must be started.

### Procedure:

Use one of the following methods to suspend the Capture program while it is running:

#### Replication Center

In the Replication Center, use the Capture Operations window to suspend the Capture program. See the Replication Center help for details.

#### asncmd suspend system command

See Chapter 17, “System commands for replication (UNIX, Windows, z/OS)” on page 303.

If you stop or suspend the Capture program during pruning, pruning is also suspended. When you resume or restart the Capture program, pruning resumes based on the **autoprune** parameter.

---

## Resuming Capture (UNIX, Windows, z/OS)

You must resume a suspended Capture program if you want it to start capturing data again.

### Prerequisites:

The Capture program with the specific Capture schema must be suspended.

### Procedure:

Use one of the following methods to resume the Capture program if it is suspended:

### **Replication Center**

In the Replication Center, use the Capture Operations window to resume a suspended Capture program. See the Replication Center help for details.

### **asnccmd resume system command**

See Chapter 17, “System commands for replication (UNIX, Windows, z/OS)” on page 303.

If you stop or suspend the Capture program during pruning, pruning is also suspended. When you resume or restart the Capture program, pruning resumes based on the **autoprun** parameter.

---

## **Reinitializing Capture**

Reinitialize the Capture program if you change any attributes of existing registered objects while the Capture program is running. For example, if you change the CONFLICT\_LEVEL, CHGONLY, RECAPTURE, CHG\_UPD\_TO\_DEL\_INS values in the register (IBMSNAP\_REGISTER) table.

For Capture on OS/400, reinitialize is also needed to start capturing data for a journal that was not being captured previously.

### **Prerequisites:**

The Capture program with the specific Capture schema must be started.

### **Procedure:**

Use one of the following methods to reinitialize the Capture program while its running:

### **Replication Center**

In the Replication Center, use the Capture Operations window to reinitialize the Capture program. See the Replication Center help for details.

### **asnccmd reinit system command**

See Chapter 17, “System commands for replication (UNIX, Windows, z/OS)” on page 303.

### **INZDPRCAP system command**

See “INZDPRCAP: Reinitializing DPR Capture (OS/400)” on page 412.

### **Related tasks:**

- Chapter 19, “Operating the replication programs (z/OS)” on page 453

- Chapter 20, “Using the Windows Service Control Manager to issue system commands (Windows)” on page 459

**Related reference:**

- “asnsrct: Creating a DB2 Replication Service to start Capture, Apply, or the Replication Alert Monitor (Windows only)” on page 338
- “asnccmd: Operating Capture (UNIX, Windows, z/OS)” on page 322
- “asnccap: Starting Capture (UNIX, Windows, z/OS)” on page 316
- “ENDDPRCAP: Stopping Capture (OS/400)” on page 400
- “STRDPRCAP: Starting Capture (OS/400)” on page 436



---

## Chapter 10. Operating the Apply program

This chapter describes how to start and stop the Apply program. It also describes how to use the ASNDONE and ASNLOAD exit routines.

This chapter contains the following sections:

- “Starting the Apply program (UNIX, Windows, z/OS)”
- “Starting the Apply program (OS/400)” on page 149
- “Stopping the Apply program” on page 151
- “Modifying the ASNDONE exit routine (UNIX, Windows, z/OS)” on page 151
- “Modifying the ASNDONE exit routine (OS/400)” on page 152
- “Refreshing target tables using the ASNLOAD exit routine” on page 154

---

### Starting the Apply program (UNIX, Windows, z/OS)

You can start an instance of the Apply program to begin applying data to your targets.

After you start the Apply program, it runs continuously (unless you used the **copyonce** start-up parameter) until one of the following events occurs:

- You stop the Apply program using the Replication Center or a command.
- The Apply program cannot connect to the Apply control server.
- The Apply program cannot allocate memory for processing.

See “Checking for current status of replication programs (UNIX, Windows, z/OS)” on page 161 to learn how you can query the status of the Apply program.

#### Prerequisites:

Before you start the Apply program, follow the instructions in Chapter 2, “Setting up for replication” on page 15 to ensure that your system is set up correctly:

- Connections are configured to all necessary replication servers.
- You have the proper authorization.
- The control tables that contain the source and control data for the desired Apply qualifier are created.
- The replication programs are configured.

- On z/OS, you manually bound the Apply program to all necessary servers.
- A password file exists for end-user authentication for remote servers running on UNIX and Windows.

Also, make sure that the following conditions are met:

- At least one active subscription set exists for the Apply qualifier and that the subscription set contains one or more of the following items:
  - Subscription-set member
  - SQL statement
  - Procedure
- All condensed target tables must have a target key, which is a set of unique columns, either a primary key or unique index, that the Apply program uses to track which changes it replicates during each Apply cycle. (Non-condensed CCD tables do not have primary keys or unique indexes.)

### **Procedure:**

Use one of the following methods to start the Apply program:

#### **Replication Center**

Use the Start Apply window. See the Replication Center help for details.

#### **asnapply system command (UNIX, Windows, z/OS)**

See “asnapply: Starting Apply (UNIX, Windows, z/OS)” on page 308 for details.

#### **Windows Services (Windows)**

See Chapter 20, “Using the Windows Service Control Manager to issue system commands (Windows)” on page 459 for details.

#### **MVS console or TSO (z/OS)**

See Chapter 19, “Operating the replication programs (z/OS)” on page 453 for details.

Regardless of which procedure you use to start the Apply program, you must set up the startup parameters. The following sections discuss the start-up parameters and recommend when to choose one value over another for each parameter.

- “apply\_path (UNIX, Windows, z/OS)” on page 141
- “apply\_qual (UNIX, Windows, z/OS)” on page 141
- “control\_server (UNIX, Windows, z/OS)” on page 142
- “copyonce (UNIX, Windows, z/OS)” on page 142
- “db2\_subsystem (z/OS)” on page 143
- “delay (UNIX, Windows, z/OS)” on page 143

- “errwait (UNIX, Windows, z/OS)” on page 143
- “inamsg (UNIX, Windows, z/OS)” on page 144
- “loadxit (UNIX, Windows, z/OS)” on page 144
- “logreuse (UNIX, Windows, z/OS)” on page 144
- “logstdout (UNIX, Windows, z/OS)” on page 145
- “notify (UNIX, Windows, z/OS)” on page 145
- “opt4one (UNIX, Windows, z/OS)” on page 145
- “pwdfile (UNIX, Windows)” on page 146
- “sleep (UNIX, Windows, z/OS)” on page 146
- “spillfile (UNIX, Windows, z/OS)” on page 147
- “sqlerrorcontinue (UNIX, Windows)” on page 147
- “term (UNIX, Windows, z/OS)” on page 148
- “trlreuse (UNIX, Windows, z/OS)” on page 148

### **apply\_path (UNIX, Windows, z/OS)**

**Default (UNIX, Windows, z/OS):** `apply_path=current_directory`

**Default (service on Windows NT):** `apply_path=sqllib\bin`

The Apply path is the directory where the Apply program stores its log and work files. By default, the Apply path is the directory where you start the program. You can change the Apply path to store the log and work files elsewhere (for example `/home/db2inst/apply_files` on an AIX system). Keep track of what directory you choose because you might need to go to this directory to access the Apply log file.

**Important:** Make sure that the directory that you choose has enough space for the temporary files used by the Apply program. For details, see “Planning space requirements for spill files for the Apply program” on page 10).

**Starting instances of Apply on one Windows NT system:** When you start the Apply program using either the Replication Center or the **asnapply** command, you *must* specify the Apply path *if* you have two or more Apply qualifiers that are identical except for their capitalization. File names on Windows NT systems are not case-sensitive. For example, assume that you have three Apply qualifiers: `APPLYQUAL1`, `ApplyQual1`, `applyqual1`. Each of these Apply instances must be started with a different **apply\_path** to prevent file name conflicts of the log files for each instance of the Apply program.

### **apply\_qual (UNIX, Windows, z/OS)**

You must specify the Apply qualifier for the subscription sets that you want to process. (You defined the Apply qualifier when you created your subscription set.) You can specify only one Apply qualifier per start command.

**Important:** The Apply qualifier is case-sensitive and the value that you enter must match the value of the APPLY\_QUAL column in the subscription sets (IBMSNAP\_SUBS\_SET) table.

If you have more than one Apply qualifier defined, you can start another instance of the Apply program. Each instance of the Apply program that you start will process different subscription sets that are represented in the same Apply control server. For example, assume that you have two subscription sets defined and each set has a unique Apply qualifier: APPLY1 and APPLY2. You can start two instances of the Apply program (one for each Apply qualifier), and each instance uses the control tables on the Apply control server called CNTRLSVR. Each instance of Apply processes its own subscription sets independently, providing better performance than if a single instance of Apply processes all the sets.

### **control\_server (UNIX, Windows, z/OS)**

**Default (UNIX, Windows):** The value of the DB2DBDFT environment variable, if available

**Default (z/OS):** None

The Apply control server is the server on which the subscription definitions and the Apply control tables reside. Specify only one control server per Apply qualifier. If you do not specify a value, the Apply program will start on the default server. The default depends on your operating system.

If the Apply program cannot connect to the control server, the Apply program terminates. If it can't connect to other servers, it does not terminate. In this case it issues an error message and continues processing.

### **copyonce (UNIX, Windows, z/OS)**

**Default:** copyonce=n

The **copyonce** parameter determines the copy cycle for the Apply program.

When you start the Apply program using **copyonce=y**, it processes each eligible subscription set only once, and then it terminates. In this case, a subscription set is eligible to be processed if one of the following conditions is met:

- The subscription set uses relative timing, the time has elapsed, and the subscription set is activated.
- The subscription set uses event-based timing, it is activated, and the event has occurred but the Apply program hasn't processed the subscription set yet.

Typically you want to start the Apply program using **copyonce=n** because you want the Apply program to continue running and processing eligible subscriptions.

If you are running the Apply program from a dial-in environment that is occasionally connected to the network, use **copyonce=y** instead of **copyonce=n**. You might also want to use **copyonce=y** if you are running the Apply program in a test environment.

**Tip:** Use **sleep=n** instead of **copyonce=y** if you want the Apply program to process each subscription set multiple times, as long as the set is eligible and data is available for replication. **Copyonce=y** processes each set only once even if there is more data to replicate.

### **db2\_subsystem (z/OS)**

The **db2\_subsystem** parameter specifies the name of the DB2 subsystem, if the **control\_server** is on z/OS. The DB2 subsystem name that you enter can be a maximum of four characters. There is no default for this parameter. This parameter is required.

### **delay (UNIX, Windows, z/OS)**

**Default:** **delay=6** seconds

The **delay** parameter sets an amount of time in seconds that the Apply program waits at the end of the Apply cycle.

By default, during continuous replication (that is, when your subscription set uses **sleep=0** minutes), the Apply program waits 6 seconds after a subscription set is processed successfully before retrying the subscription set. Use a non-zero delay value to save CPU cycles when there is no database activity to be replicated. Use a lower delay value for low latency.

### **errwait (UNIX, Windows, z/OS)**

**Default:** **errwait=300** seconds (5 minutes)

The **errwait** parameter specifies the number of seconds that the Apply program waits before retrying a subscription set after a subscription cycle failed

By default, the Apply program waits 300 seconds before it retries a subscription set after a subscription cycle failed. You might want to use a smaller value in a test environment. The minimum value is 1 second. In a production environment, consider the trade-offs before you change the default for this parameter:

- If you use a smaller value, you might waste CPU cycles if the Apply program keeps retrying hard errors. For example, you will use CPU cycles

unnecessarily if the Apply program keeps retrying to process a subscription set when there is a problem with a target table. You might get an undesirably large number of messages in the log file and, if the Apply program runs on z/OS, on the operator console.

- If you use a larger value, you might increase latency if the Apply program must wait to retry transient error conditions. For example, you will increase latency if you use a larger value for the **errwait** parameter because the Apply program waits unnecessarily after it encounters a network error that might be corrected quickly.

### **inamsg (UNIX, Windows, z/OS)**

**Default:** **inamsg=y**

The **inamsg** parameter specifies whether or not the Apply program issues a message when it becomes inactive.

By default, the Apply program issues a message when it becomes inactive. You might not want the Apply program to issue a message when it becomes inactive because the messages will take up a lot of space in the Apply log file, especially if the Apply program is not waiting long between processing subscription sets. To turn off these messages, use **inamsg=n**.

### **loadxit (UNIX, Windows, z/OS)**

**Default:** **loadxit=n**

The **loadxit** parameter specifies whether or not the Apply program should refresh target tables using the ASNLOAD exit routine.

By default, the Apply program does not use the ASNLOAD exit routine to refresh target tables (**loadxit=n**). Use **loadxit=y** if you want the Apply program to invoke the ASNLOAD exit routine to refresh target tables. Consider using the ASNLOAD exit if there is a large amount of data to be copied to the target tables during a full refresh. For information about when you might want the Apply program to use ASNLOAD and for instructions on how to use ASNLOAD, see “Refreshing target tables using the ASNLOAD exit routine” on page 154.

### **logreuse (UNIX, Windows, z/OS)**

**Default:** **logreuse=n**

The Apply program stores operational information in a log file. On UNIX and Windows, the name of the log file is *db2instance.control\_server.apply\_qualifier.APP.log*. On the z/OS operating system, the file name is similar except it does not contain the DB2 instance name.

The parameter specifies whether to append to the log file or to overwrite it.

By default, the Apply program appends messages to the log file (**logreuse=n**) each time that you start the Apply program. Keep the default if you want the history of the messages that are issued by the Apply program. In the following situations you might want to use **logreuse=y**, where the Apply program deletes the log and re-creates the log when it starts:

- The log is getting large, and you want to clean out the log to save space.
- You don't need the history that is stored in the log.

## **logstdout (UNIX, Windows, z/OS)**

**Default:** **logstdout=n**

The **logstdout** parameter is available only if you use the **asnapply** command, it is not available in the Replication Center.

The **logstdout** parameter specifies whether the Apply program sends messages to the log file only or to the log file and to standard output.

By default, the Apply program sends messages to the log file only. You might choose to send messages to standard output (**logstdout=y**) if you are troubleshooting or monitoring how your Apply program is operating.

## **notify (UNIX, Windows, z/OS)**

**Default:** **notify=n**

The **notify** parameter specifies whether the Apply program notifies the ASNDONE exit routine after it processes a subscription.

By default, the Apply program does not notify the ASNDONE exit routine after subscription processing completes. If you specify **notify=y**, after the Apply program completes a subscription cycle it invokes ASNDONE to perform additional processing, such as examining the Apply control tables or sending e-mail messages. For more information about ASNDONE, see “Modifying the ASNDONE exit routine (UNIX, Windows, z/OS)” on page 151.

## **opt4one (UNIX, Windows, z/OS)**

**Default:** **opt4one=n**

The **opt4one** parameter specifies whether or not the Apply program processing is optimized for one subscription set.

By default, the Apply program is optimized for many subscription sets. The Apply program reads the information from the replication control tables at the beginning of each copy cycle. If you have one subscription set for the Apply

qualifier, start the Apply program using **opt4one=y** so that the Apply program caches in memory information about the subscription set members and columns and reuses it. When you optimize the Apply program for one subscription set, the Apply program uses less CPU, and you improve throughput rates.

**Important:** When you use **opt4one=y** and you add a member to a set or otherwise modify a set, you must stop the Apply program and start it again so that the Apply program picks up the changes in the control tables.

### **pwdfile (UNIX, Windows)**

**Default:** **pwdfile=asnpwd.aut**

If your data is distributed across servers, you can store user IDs and passwords in an encrypted password file so that the Apply program can access data on remote servers. For more information, see “Storing user IDs and passwords for replication (UNIX, Windows)” on page 23.

### **sleep (UNIX, Windows, z/OS)**

**Default:** **sleep=y**

The **sleep** parameter specifies whether the Apply program continues running in sleep mode or terminates after it processes eligible subscription sets.

By default, the Apply program starts with **sleep=y**. It checks for eligible subscription sets. If it finds an eligible subscription set, it processes it and continues looking for another eligible set. Apply continues to process eligible sets if it finds them. When it cannot find any more eligible sets, the Apply program continues running in sleep mode and it “wakes up” periodically to check if any subscription sets are eligible. Usually you want to start the Apply program in this way because you want updates applied over time and you expect the Apply program to be up and running.

When you start the Apply program with **sleep=n**, the Apply program checks for eligible subscription sets and processes them. It continues processing eligible subscription sets until it can’t find any more eligible sets, and it repeats the process for eligible sets until there is no more data to replicate; then, the Apply program terminates. Typically you want to use **sleep=n** in a mobile environment or in a test environment where you want the Apply program to run *only* if it finds eligible subscription sets, and then you want it to terminate. You don’t want the Apply program to wait in sleep mode and wake up periodically to check for more eligible sets. In these environments you want to control when Apply runs rather than have it run endlessly.

**Tip:** Use **copyonce=y** instead of **sleep=n** if you want to process each subscription set only once.



## **spillfile (UNIX, Windows, z/OS)**

**Default (UNIX, Windows):** `spillfile=disk`

**Default (z/OS):** `spillfile=MEM`

Apply retrieves data from the source tables and places it in a spill file on the system where the Apply program is running.

On UNIX and Windows operating systems, the only valid setting for `spillfile` is `disk` because spill files are *always* on disk in the location specified by the `apply_path` parameter.

On z/OS operating systems, including USS, the spill file is stored in memory by default. You can specify to store it on disk, such that the Apply program uses the specifications on the ASNAPLDD card to allocate spill files. If the ASNAPLDD card is not specified, it uses VIO. If your replication cycles are short, it's appropriate to create spill files in memory if the amount of data being replicated is small. See "Planning space requirements for spill files for the Apply program" on page 10 for details.

## **sqlerrorcontinue (UNIX, Windows)**

**Default:** `sqlerrorcontinue=n`

The `sqlerrorcontinue` parameter specifies how the Apply program should react to certain SQL errors.

By default, when the Apply program encounters any SQL error, it stops processing that subscription set and generates an error message. Typically you would use the default in your production environment.

If you are in a test environment, you can expect certain SQL errors to occur when inserting data into target tables. Sometimes those errors are acceptable to you, but they would cause the current subscription cycle to stop. In those situations, you can start the Apply program using `sqlerrorcontinue=y` so that it ignores those errors and does not rollback replicated data from that cycle. If the Apply program receives an SQL error when inserting data into a target table, it checks the values in the `apply_qualifier.sqs` file. If it finds a match, it writes the details about the error to an error file, `apply_qualifier.err`, and it continues processing. If the Apply program encounters an SQL error that is not listed in the `apply_qualifier.sqs` file, it stops processing the set and goes on to the next set.

Before you start the Apply program using the `sqlerrorcontinue=y` option, you must create the `apply_qualifier.sqs` file and store it in the directory from which you invoke the Apply program. List up to 20 five-byte values, one after the

other, in the file. If you change the contents of this file when the Apply program is running, stop the Apply program and start it again so that it recognizes the new values.

**Example:** Assume that you want the Apply program to continue processing a subscription set if a target table gets the following error (sqlstate/code):

**42704/-803**

Duplicate index violation

You would create an SQL state file that contains the following SQL state:

42704

If the SQL state is returned when updating the target table, the Apply program applies the changes to the other target tables within the set and creates an error file indicating both the error and the rejected rows.

**Tip:** Check the STATUS column of the Apply trail (IBMSNAP\_APPLYTRAIL) table. A value of 16 indicates that the Apply program processed the subscription set successfully, but some of the allowable errors, which you defined in the *apply\_qualifier.sqs* file, occurred.

## **term (UNIX, Windows, z/OS)**

**Default:** term=y

The **term** parameter determines how the status of DB2 affects the operation of the Apply program.

By default, the Apply program terminates if DB2 terminates.

Use **term=n** if you want the Apply program to wait for DB2 to start, if DB2 is not active. On the z/OS platform, if DB2 quiesces and Apply is active, the Apply program will stay active and will not reconnect until DB2 is started again. On UNIX and Windows platforms, if DB2 quiesces and Apply is active, the Apply program will stay active and will not reconnect until DB2 is out of quiesce mode.

## **trlreuse (UNIX, Windows, z/OS)**

**Default:** trlreuse=n

The **trlreuse** parameter specifies whether or not the Apply trail (IBMSNAP\_APPLYTRAIL) table should be reused (appended to) or overwritten when the Apply program starts.

By default, when the Apply program starts, it appends entries to the Apply trail table. This table contains the history of operations for all Apply instances at the Apply control server. It is a repository of diagnostic and performance

statistics. Keep the default if you want the history of updates. In the following situations you might want the Apply program to empty the Apply trail table when it starts instead of appending to it (**trlreuse=y**):

- The Apply trail table is getting too large, and you want to clean it out to save space.
- You don't need the history that is stored in the table.

**Tip:** Instead of using **trlreuse=y**, you can use SQL processing after the Apply program successfully completes a subscription set (where **status=0**) to delete rows from the Apply trail table.

---

## Starting the Apply program (OS/400)

You can start an instance of the Apply program to begin applying data to your targets.

After you start the Apply program, it runs continuously unless one of the following conditions are true:

- You started the program using the COPYONCE(\*YES) start-up parameter.
- You specified ALWINACT(\*NO) and there is no data to be processed.
- You stop the Apply program using the Replication Center or a command.
- The Apply program cannot connect to the Apply control server.
- The Apply program cannot allocate memory for processing.

### Prerequisites:

Before you start the Apply program, ensure that your system is set up correctly:

- Connections are configured to all replication servers.
- You have the proper authorization.
- The control tables are created.
- The replication programs are configured.

Also, make sure that the following conditions are met:

- At least one active subscription set exists for the Apply qualifier and that subscription set contains one or more of the following items:
  - Subscription-set member
  - SQL statement
  - Procedure
- All condensed target tables must have a target key, which is a set of unique columns, either a primary key or unique index, that the Apply program

uses to track which changes it replicates during each Apply cycle.  
(Non-condensed CCD tables do not have primary keys or unique indexes.)

**Procedure:**

Use one of the following methods to start the Apply program:

**Replication Center**

Use the Start Apply window. See the Replication Center help for details.

**STRDPRAPY system command**

See “STRDPRAPY: Starting Apply (OS/400)” on page 427 for details.

When you start the Apply program, you can use these default settings for the operational parameters:

*Table 7. Default settings for the Apply program (OS/400)*

Operational parameter	Description of (*value)
USER (*CURRENT)	The user who signed on to the system.
JOB (*LIBL/QZSNDPR)	Product library name / job description.
APYQUAL (*USER)	Current user name (from above).
CTLSVR (*LOCAL)	Local RDB server name.
TRACE (*NONE)	Do not generate a trace.
FULLREFPGM (*NONE)	Do not run the ASNLOAD exit routine.
SUBNFYPGM (*NONE)	Do not run the ASNDONE exit routine.
INACTMSG (*YES)	When the Apply program begins an inactive period, it generates message ASN1044 which describes how long the program will be inactive.
ALWINACT (*YES)	Sleep if there is nothing to process.
DELAY (6)	Wait 6 seconds after an Apply cycle before processing again.
RTYWAIT (300)	Wait 300 seconds before retrying a failed operation.
COPYONCE (*NO)	Do not terminate after completing one copy cycle, continue processing.
TRLREUSE (*NO)	Do not empty the Apply trail (IBMSNAP_APPLYTRAIL) table when the Apply program starts.
OPTSNGSET (*NO)	Do not optimize performance of the Apply program for processing a single subscription set.

For more information about these operational parameters, including syntax diagrams, see “STRDPAPY: Starting Apply (OS/400)” on page 427.

---

## Stopping the Apply program

You can stop an instance of the Apply program. When you stop the Apply program, it no longer copies data to the target tables, and it updates information in the control tables to ensure that the program starts cleanly the next time that you start it.

### Prerequisites:

The instance of the Apply program must be started.

### Procedure:

Use one of the following methods to stop an instance of the Apply program:

#### Replication Center

Use the Stop Apply window. See the Replication Center help for details.

#### asnacmd stop system command (UNIX, Windows, z/OS)

See “asnacmd: Operating Apply (UNIX, Windows, z/OS)” on page 304 for details.

#### ENDDPRAPY system command (OS/400)

See “ENDDPRAPY: Stopping Apply (OS/400)” on page 397 for details.

---

## Modifying the ASNDONE exit routine (UNIX, Windows, z/OS)

This section describes how to customize the ASNDONE exit routine on UNIX, Windows, and z/OS operating systems.

If you start the Apply program with the **notify=y** parameter, the Apply program calls the ASNDONE exit routine after it finishes processing subscriptions, regardless of whether the subscriptions were processed successfully. The following list describes some examples of how you might modify the ASNDONE exit routine to use it in your replication environment:

- Use the exit routine to examine the UOW table for rejected transactions and initiate further actions (for example, send e-mail automatically to the replication operator, issue a message, or generate an alert) if a rejected transaction is detected.
- Use the exit routine to deactivate a failed subscription set so that the Apply program avoids retrying that subscription set until it is fixed. To detect a failed subscription set, modify the exit routine to look for STATUS= -1 in the Apply trail (IBMSNAP\_APPLYTRAIL) table. To deactivate the

subscription set, configure the exit routine so that it sets `ACTIVATE=0` in the subscription sets (`IBMSNAP_SUBS_SET`) table.

- Use the exit routine to manipulate data after it is applied for *each* subscription set. (Alternatively, you can define run-time processing statements using SQL statements or stored procedures that run before or after the Apply program processes a *specific* subscription set.)

#### Procedure:

To use a modified version of the `ASNDONE` sample exit routine:

1. Modify the `ASNDONE` routine to meet your requirements.  
**UNIX, Windows:** See the PROLOG section of the sample program (`\sqllib\samples\repl\asndone.smp`) for information about how to modify this exit routine.  
**z/OS:** See the PROLOG section of the sample program `SASNAJCL(ASNDONE)`.
2. Compile, link, and bind the program and place the executable in the appropriate directory.
3. Start the Apply program with the `notify=y` parameter to call the `ASNDONE` exit routine.

---

## Modifying the `ASNDONE` exit routine (OS/400)

This section describes how to customize the `ASNDONE` exit routine for an OS/400 environment.

If you start the Apply program with the `SUBNFYPGM` parameter set to the name of the `ASNDONE` exit routine, the Apply program calls the `ASNDONE` exit routine after it finishes processing subscriptions, regardless of whether the subscriptions were processed successfully. The following list describes some examples of how you might modify the `ASNDONE` exit routine to use it in your replication environment:

- Use the exit routine to examine the UOW table for rejected transactions and initiate further actions (for example, send e-mail automatically to the replication operator, issue a message, or generate an alert) if a rejected transaction is detected.
- Use the exit routine to deactivate a failed subscription set so that the Apply program avoids retrying that subscription set until it is fixed. To detect a failed subscription set, modify the exit routine to look for `STATUS= -1` in the Apply trail (`IBMSNAP_APPLYTRAIL`) table. To deactivate the subscription set, configure the exit routine so that it sets `ACTIVATE=0` in the subscription sets (`IBMSNAP_SUBS_SET`) table.
- Use the exit routine to manipulate data after it is applied for *each* subscription set. (Alternatively, you can define run-time processing

statements using SQL statements or stored procedures that run before or after the Apply program processes a *specific* subscription set. See “Enhancing data using stored procedures or SQL statements” on page 112 for details.)

### Procedure:

To use a modified version of the ASNDONE sample exit routine:

1. Modify the ASNDONE exit routine to meet your site’s requirements.

The following table indicates where you can find the source code for this routine in C, COBOL, and RPG languages:

Compiler language	Library name	Source file name	Member name
C	QDP4	QCSRC	ASNDONE
COBOL	QDP4	QCBLLSRC	ASNDONE
RPG	QDP4	QRPGLSRC	ASNDONE

When modifying the program, consider these activation group concerns:

**If the program is created to run with a new activation group:** The Apply program and the ASNLOAD program will not share SQL resources, such as relational database connections and open cursors. The activation handling code in the OS/400 operating system frees any resources allocated by the ASNLOAD program before control is returned to the Apply program. Additional resource is used every time that the Apply program calls the ASNLOAD program.

**If the program is created to run in the caller’s activation group:** It shares SQL resources with the Apply program. Design the program so that you minimize its impact on the Apply program. For example, the program might cause unexpected Apply program processing if it changes the current relational database connection.

**If the program is created to run in a named activation group:** It does not share resources with the Apply program. Use a named activation group to avoid the activation group overhead every time the ASNLOAD program is called. Run-time data structures and SQL resources can be shared between invocations. Application clean-up processing is not performed until the Apply program is ended, so design the subscription notify program to ensure that it does not cause lock contention with the Apply program by leaving source tables, target tables, or control tables locked when control is returned to the Apply program.

2. Compile, link, and bind the program, and place the executable in the appropriate directory.

3. Start the Apply program and specify the name of the ASNDONE program using the parameter SUBNFYPGM on the **STRDPRAPY** command. For example, if the program is named ASNDONE\_1 and resides in library APPLIB, use the following command:

```
SUBNFYPGM(APPLIB/ASNDONE_1)
```

---

## Refreshing target tables using the ASNLOAD exit routine

By default, the Apply program does not use the ASNLOAD exit routine when it performs a full refresh for each target table in a subscription set. It does a full select against the source table, brings the data to a spill file on the server where the Apply program is running, and uses INSERT statements to populate the target table. If you have large source tables, you might want to use the ASNLOAD exit routine instead to copy the data more efficiently to the target during a full refresh.

The ASNLOAD exit routine is shipped as a sample exit program in both a source format and a compiled format. The sample exit program differs on each DB2 platform, in each case taking advantage of the utility options offered on that platform.

If an error occurs when the Apply program calls the ASNLOAD exit routine, or if the exit routine returns a nonzero return code, the Apply program issues a message, stops processing the current subscription set, and processes the next subscription set.

### Prerequisites:

Ensure that the following prerequisites are met before you use the ASNLOAD exit routine:

- The target-table columns match both the order *and* data type of the source tables.
- The target table contains only columns that are part of the replication mapping.

The following sections describe how to use the ASNLOAD exit routine on various platforms:

- “Refreshing target tables with the ASNLOAD exit routine (UNIX, Windows)” on page 155
- “Refreshing target tables with the ASNLOAD exit routine (z/OS)” on page 156
- “Refreshing target tables with the ASNLOAD exit routine (OS/400)” on page 159



## Refreshing target tables with the ASNLOAD exit routine (UNIX, Windows)

The ASNLOAD exit routine offers many utility options, such as using the DB2 EXPORT utility with either the DB2 IMPORT utility or the DB2 LOAD utility, or using the new LOAD FROM CURSOR utility. When you invoke the sample exit routine, by default it chooses which utility to use based on the source server, target server, and run-time environment.

You can use the compiled exit routine, you can configure its behavior by customizing the replication configuration, or you can customize the exit code itself. You can customize the replication configuration by either updating columns in the subscription members (IBMSNAP\_SUBS\_MEMBR) table or by updating a sample configuration file (asnload.ini).

### Procedure:

To use the ASNLOAD routine as provided, start the Apply program using the **loadxit=y** parameter.

To use a modified version of the ASNLOAD sample exit routine:

1. Modify the ASNLOAD routine to meet your site's requirements. See the PROLOG section of the sample program (\sqlib\samples\repl\asnload.smp) for information about how to modify this exit routine.

**Important:** The sample source uses user ID and password combinations from the asnload.ini file. If the asnload.ini file does not have a user ID and password for a particular server, or if the asnload.ini file is not available, connect without the user/using phrases will be made.

2. Compile, link, and bind the program and place the executable in the appropriate directory.
3. Set LOADX\_TYPE to 2 for members that will be populated using the code you provide. For more information, see "Customizing ASNLOAD exit behavior (UNIX, Windows, z/OS)" on page 157.
4. Start the Apply program with the **loadxit=y** parameter to call the ASNLOAD exit routine.

To configure input to the ASNLOAD exit routine, see "Using the configuration file for ASNLOAD" on page 159.

### Files generated by the ASNLOAD exit routine:

These files are stored in the **apply\_path** directory for the Apply instance that invoked the ASNLOAD exit routine.

- `asnload | | apply_qualifier.trc`

This file contains trace information if the trace is turned on. The ASNLOAD exit routine creates this file. If the file exists, information is appended to the file.

- asnload | | *apply\_qualifier.msg*

This file contains general exit failure, warning, and informational messages, including load statistics. The ASNLOAD exit routine creates this file. If the file exists, information is appended to the file.

- asnaEXPT | | *apply\_qualifier.msg*

This file contains error, warning, or informational messages issued by the DB2 EXPORT utility. The ASNLOAD exit routine creates this file. If the file exists, information is appended to the file.

- asnaIMPT | | *apply\_qualifier.msg*

This file contains error, warning, or informational messages issued by the DB2 IMPORT utility. The ASNLOAD exit routine creates this file. If the file exists, information is appended to the file.

- asnaLOAD | | *apply\_qualifier.msg*

This file contains error, warning, or informational messages issued by the DB2 LOAD utility. The ASNLOAD exit routine creates this file. If the file exists, information is appended to the file.

## Refreshing target tables with the ASNLOAD exit routine (z/OS)

The ASNLOAD exit routine calls the LOAD utility, which does cursor-based fetches to get data from the source and loads the data to the target. The ASNLOAD exit routine uses LOAD with LOG NO and resets the COPYPEND status of the table space. You can modify the sample ASNLOAD source code to change the load options. The source consists of two header files and three C++ programs.

### Procedure:

To use the ASNLOAD routine as provided, start the Apply program using the **loadxit=y** parameter.

To use a modified version of the ASNLOAD sample routine:

1. Modify the ASNLOAD routine to meet your site's requirements. See the PROLOG section of the sample program SASNAJCL(ASNLOAD) for information about how to modify this exit routine.
2. Compile, link, and bind the program and place the executable in the appropriate directory. Add the ASNLOAD package to the Apply plan.
  - a. Make sure that the following conditions are met:
    - DB2 Universal Database for z/OS and OS/390 Version 7 or later, with utility support, is installed.

- DSNUTILS stored procedure is running. DSNUTILS must run in a WLM environment. For more information about using DSNUTILS, see the *DB2 Universal Database for OS/390 and z/OS Utility Guide and Reference*, SC26-9945 for more information.
  - b. Precompile the two header files and three programs with the connect(1) precompiler option. Sample JCL to precompile, compile, link, and bind is provided with the sample ASNLOAD programs.
  - c. Linkedit the ASNLOAD exit routine. You must compile and link with SETCODE AC(1) in case the Apply program runs from an APF-authorized library.
  - d. Bind ASNLOAD exit routine with DSNUTILS and the Apply package. The sample ASNLOAD runs load with LOG NO and then repairs the table space to set nocopypend. It does not back up the table spaces. ASNLOAD creates temporary files under the user ID running the instance of Apply. It also creates a file that contains all the information regarding the load.
3. Set loadx\_type = 2 for members that will be populated using the code you provided.
  4. Start the Apply program with the **loadxit=y** parameter to call the ASNLOAD exit routine.

To configure input to the ASNLOAD exit routine, see “Using the configuration file for ASNLOAD” on page 159.

**Files generated by the ASNLOAD exit routine:** These files are stored in the **apply\_path** directory or HLQ for the Apply instance that invoked the ASNLOAD exit routine.

- *userid.apply\_qual.LOADMSG*  
This file contains failure, warning, and informational messages, including load statistics. The ASNLOAD exit routine creates this file. If the file exists, information is appended to the file.
- *userid.apply\_qual.LOADTRC*  
This file contains trace information if the trace is turned on. The ASNLOAD exit routine creates this file. If the file exists, information is appended to the file.

## Customizing ASNLOAD exit behavior (UNIX, Windows, z/OS)

You can configure the behavior of the ASNLOAD exit routine by customizing the replication configuration, or you can customize the exit code itself. You can customize the replication configuration by either updating columns in the subscription member (IBMSNAP\_SUBS\_MEMBR) table or by updating a configuration file.

### Using the subscription members table

You can use columns in the subscription member (IBMSNAP\_SUBS\_MEMBR) table to customize the behavior of the ASNLOAD exit routine. Use LOADX\_TYPE to select the load option. The valid values for LOADX\_TYPE are:

#### **null (default)**

On z/OS: Use the crossloader utility.

On UNIX, Windows: The ASNLOAD exit routine determines the most appropriate utility (option 3, 4, or 5).

#### **1 Do not call ASNLOAD exit routine for this member.**

Set LOADX\_TYPE to 1 if you do not want the ASNLOAD exit routine to be called for that member.

#### **2 Provide your own exit logic.**

If you want to provide your own logic in the ASNLOAD exit routine, set LOADX\_TYPE to 2 for those subscription set members that you want populated by the ASNLOAD exit routine. For example, you might provide a mechanism within the sample exit routine to invoke the Sybase bulk copy program (bcp) to load a Sybase target table after exporting data from a DB2 database in a bcp-ready format using the bcp option in the DB2 Export utility. If you set LOADX\_TYPE to 2 but you do not provide exit logic, the exit will fail.

#### **3 Use the crossloader utility.**

On UNIX and Windows platforms the crossloader utility requires a SELECT statement to fetch the data that is to be loaded to the target. This statement can refer either to a DB2 table that is local to the target table being loaded, or it can refer to a remote table through a nickname. If the replication source is a DB2 table that is remote to the target table, you must create a nickname for the DB2 source table at the target server database. Also, you must supply the nickname owner and the table in the LOADX\_SRC\_N\_OWNER and LOADX\_SRC\_N\_TABLE columns of the subscription member (IBMSNAP\_SUBS\_MEMBR) table. For non-DB2 replication sources, the replication control tables already contain a nickname that can be used by the utility, and you do not need to add information to LOADX\_SRC\_N\_OWNER and LOADX\_SRC\_N\_TABLE.

#### **4 (UNIX and Windows only)**

Use a combination of the EXPORT utility and the LOAD utility.

#### **5 (UNIX and Windows only)**

Use a combination of the EXPORT utility and the IMPORT utility.

## Using the configuration file for ASNLOAD

You can use an optional configuration file to configure input to the ASNLOAD exit routine. This file is not required for ASNLOAD to run.

On UNIX, and Windows platforms, the configuration file must have the file name `asnload.ini`. The ASNLOAD exit routine looks for this optional configuration file in the **apply\_path** directory. Edit the sample file `sqllib/samples/repl/asnload.ini` and store it in the **apply\_path** directory for the Apply instance that invoked the ASNLOAD exit routine.

On the z/OS platform, the configuration file must be a cataloged sequential file named `apply_qualifer.ASNLOAD.INI`. If you want to configure input to the ASNLOAD exit routine, use the sample JCL provided to create this optional file.

## Refreshing target tables with the ASNLOAD exit routine (OS/400)

Use the exit routine instead of the Apply program to perform a full refresh more efficiently. For example, if you are copying every row and every column from a source table to a target table, you can design a full-refresh exit routine that uses a Distributed Data Management (DDM) file and the Copy File (CPYF) CL command to copy the entire file from the source table to the target table.

### Procedure:

To refresh target tables using the ASNLOAD exit routine, start the Apply program using the `FULLREFPGM` parameter.

To use a modified version of the ASNLOAD sample routine:

1. Modify the ASNLOAD exit routine to meet your site's requirements. See the PROLOG section of the sample program for information about how to modify this exit routine. The source is available in C, COBOL, and RPG languages:

Compiler language	Library name	Source file name	Member name
C	QDP4	QCSRC	ASNLOAD
COBOL	QDP4	QCBLLSRC	ASNLOAD
RPG	QDP4	QRPGLESRC	ASNLOAD

2. Compile, link, and bind the program and place the executable in the appropriate directory.

To avoid interference with the Apply program, compile the exit routine so that it uses a new activation group (not the activation group of the caller).

You can compile the exit routine with a named activation group or with a new activation group. To get better performance, use a named activation

group. With a named activation group, the exit routine must commit or roll back changes as needed. The Apply program will not cause changes to be committed or rolled back (unless it ends). The exit routine should either explicitly commit changes, or it should be compiled to implicitly commit changes when it completes. Any uncommitted changes when the exit routine completes are not committed until either:

- The Apply program calls another exit routine with the same activation group.
  - The job started for the Apply program ends.
3. Start the Apply program with the FULLREFPGM parameter set to the name of the ASNLOAD program.

When you start the Apply program, it uses the ASNLOAD exit routine that you specified. If you want it to use another ASNLOAD exit routine, end the Apply program and start it again.

When you run the ASNLOAD exit routine, it refreshes all the target tables, table by table.

**Related tasks:**

- Chapter 19, “Operating the replication programs (z/OS)” on page 453
- Chapter 20, “Using the Windows Service Control Manager to issue system commands (Windows)” on page 459

**Related reference:**

- “asnsrct: Creating a DB2 Replication Service to start Capture, Apply, or the Replication Alert Monitor (Windows only)” on page 338
- “asnacmd: Operating Apply (UNIX, Windows, z/OS)” on page 304
- “asnapply: Starting Apply (UNIX, Windows, z/OS)” on page 308
- “ENDDPRAPY: Stopping Apply (OS/400)” on page 397
- “STRDPRAPY: Starting Apply (OS/400)” on page 427

---

## Chapter 11. Monitoring replication

This chapter describes methods you can use to monitor your replication environment. Using the information in this chapter, you can check the current status of the replication programs or review historical data to determine recent messages and throughput or latency statistics. In addition, you can automatically monitor your replication environment with the Replication Alert Monitor, which runs independently of the Capture and Apply programs.

This chapter contains the following sections:

- “Checking for current status of replication programs (UNIX, Windows, z/OS)”
- “Checking the status of the Capture and Apply journal jobs (OS/400)” on page 163
- “Reviewing historical data for trends” on page 163
- “Setting up automated monitoring of your replication environment” on page 168
- “Monitoring the progress of the Capture program (OS/400)” on page 180

---

### Checking for current status of replication programs (UNIX, Windows, z/OS)

You can quickly assess the current status of the Capture program, Apply program, or Replication Alert Monitor program.

Use one of the following methods to check the current status of the replication programs:

#### Replication Center (UNIX, Windows, z/OS)

Use the Query Status window to check the current status of the Capture or Apply programs. (You cannot query the status of the Replication Alert Monitor using the Replication Center.) See the Replication Center help for details.

#### Command line (UNIX, Windows, z/OS)

- Capture program **asnccmd** system command, **status** parameter. See “asnccmd: Operating Capture (UNIX, Windows, z/OS)” on page 322 for details.
- Apply program **asnacmd** system command, **status** parameter. See “asnacmd: Operating Apply (UNIX, Windows, z/OS)” on page 304 for details.

- Replication Alert Monitor **asnmcmd** system command, **status** parameter. See “asnmcmd: Operating the Replication Alert Monitor (UNIX, Windows, z/OS)” on page 329 for details.

When you query the status of a program, you receive messages that describe the state of each thread that is associated with that program:

- The Capture program has four threads: Administration thread, pruning thread, worker thread, serialization thread.
- The Apply program has two threads: Administration thread, worker thread.
- The Replication Alert Monitor program has three threads: Administration thread, worker thread, serialization thread.

You can discern whether your programs are working correctly by the messages you receive. Typically worker threads, administration threads, and pruning threads are in a working state and are performing the tasks that they were intended to perform. Serialization threads are typically in the waiting state; they are global signal handlers and are usually waiting for signals. The pruning thread prunes the CD tables and the following replication control tables.

- Unit-of-work (IBMSNAP\_UOW) table
- Capture trace (IBMSNAP\_CAPTRACE) table
- Capture monitor (IBMSNAP\_CAPMON) table
- Signal (IBMSNAP\_SIGNAL) table

If the messages that you receive indicate that the program is working but you find evidence in your environment to the contrary, you must do more investigating. For example, if you query the status of the Apply program and you find that the worker thread is working but you notice that data is not being applied to the target tables as you expected, look in the Apply trail (IBMSNAP\_APPLYTRAIL) table for messages that might explain why the data is not being applied. Perhaps there are some system resource problems preventing the program from working.

If the messages you receive do not indicate a typical state, you might have to take further action as described in Table 8.

*Table 8. Suggested actions for problems related to status of processing threads*

Status of processing thread	Description and suggested action
Exists	The thread exists but cannot start. Contact IBM Software Support.
Was started	Investigate potential system resource problems, such as not enough CPU.



Table 8. Suggested actions for problems related to status of processing threads (continued)

Status of processing thread	Description and suggested action
Is initializing	The thread is initialized but cannot work. Contact IBM Software Support.
Is resting	This state pertains only to threads for the Capture program. If your threads are in this state, you have suspended the Capture program and it is waiting for you to resume its operations.
Is stopped	The thread is not running. Check the Apply trail (IBMSNAP_APPLYTRAIL) or Capture trace (IBMSNAP_CAPTRACE) table for messages that explain why the thread stopped. For example, if you get a message indicating that the pruning thread stopped, check the IBMSNAP_CAPTRACE table to find out why. If your tables are too big and you want to prune them now, you can stop the Capture program and start it again to start the pruning thread.

## Checking the status of the Capture and Apply journal jobs (OS/400)

On DB2 for iSeries, use the Work with Subsystem Jobs (**WRKSBSJOB**) system command to check the status of the journal jobs for the Capture and Apply programs.

1. Enter the command:

```
WRKSBSJOB subsystem
```

where *subsystem* is the subsystem name. In most cases the subsystem is QZSNDPR, unless you created your own subsystem description.

2. In the list of running jobs, look for the jobs you're interested in. The journal job is named after the journal to which it was assigned. If a job is not there, use the Work with Submitted Jobs (**WRKSBMJOB**) system command or the Work with Job (**WRKJOB**) system command to locate the job. Find the job's joblog to verify that it completed successfully or to verify why it failed.

## Reviewing historical data for trends

You can review historical data from your recent replication operations and evaluate the data for trends. The trends that you recognize over time might demonstrate to you that a steady volume of data is being replicated or might indicate that adjustments could be made to improve performance.

Historical data is derived from the following control tables: Apply trail (IBMSNAP\_APPLYTRAIL), Apply trace (IBMSNAP\_APPLYTRACE), Capture monitor (IBMSNAP\_CAPMON), and Capture trace (IBMSNAP\_CAPTRACE). The frequency with which you prune these tables affects the reports that you can generate. It is recommended that you keep at least one week of data in these tables so that you can examine that data when you are troubleshooting or evaluating performance.

Table 9 describes the historical data that you can view.

*Table 9. Where to find historical information*

<b>To answer this question:</b>	<b>Use the following Replication Center window:</b>
What are the recent messages from the Capture program?	Capture Messages
On average: <ul style="list-style-type: none"> <li>• How many rows were processed in the CD table for a given period of time?</li> <li>• How many rows were pruned?</li> <li>• How many transactions were committed?</li> <li>• How much memory is the Capture program using?</li> </ul>	Capture Throughput Analysis
On average, what is the approximate length of time between when data was updated at the source and when it was captured by the Capture program?	Capture Latency
What are the recent messages from the Apply program?	Apply Report
On average, <ul style="list-style-type: none"> <li>• How many rows were processed in the target table for a given period of time?</li> <li>• What is the elapsed time for processing of subscription sets?</li> </ul>	Apply Throughput Analysis
On average, approximately how much time has elapsed between the time when the source table was updated and when the corresponding target table was updated?	End-to-End Latency

You can select a range of time to identify how much data you want to analyze. Specify the dates and times for both the beginning and the end of a time range, and then specify that the results be displayed as an average of the calculated rates. You select intervals of time (one second, one minute, one hour, one day, or one week) to group the results. For example, if you chose to analyze the Apply throughput from 9:00 p.m. and 9:59 p.m., and you want the data displayed in one minute intervals, the results are displayed in 60 rows, each row summarizing the activity for a single minute out of the 60-minute range. Alternatively, if you chose a one-hour interval, the results are displayed in 1 row, showing the average throughput for the specified hour time period. If you don't specify an interval, you can view raw data from the APPLYTRAIL table.

The Replication Center windows show results from information contained in various control tables and log files. The following sections describe in more detail how you can use historical data to evaluate your replication operations with the Replication Center:

- “Reviewing Capture program messages”
- “Examining Capture program throughput”
- “Displaying latency of data processed by the Capture program” on page 166
- “Reviewing Apply program messages” on page 167
- “Examining Apply program throughput” on page 167
- “Displaying the average length of time taken to replicate transactions” on page 167

## **Reviewing Capture program messages**

Use the Capture Messages window to review the messages that were inserted in the Capture trace (IBMSNAP\_CAPTRACE) table over a specified period of time. The IBMSNAP\_CAPTRACE table contains a row for significant events such as initialization, pruning, warnings, and errors that are issued by the Capture program.

For example, from the Capture Messages window, you can review all the error and warning messages that are recorded by the Capture program during one week.

## **Examining Capture program throughput**

Use the Capture Throughput Analysis window to display the performance results of a Capture program over a specific range of time. The Capture program regularly records statistical information in the Capture monitor (IBMSNAP\_CAPMON) table, and during pruning it records pruning statistics in the Capture trace (IBMSNAP\_CAPTRACE) table. Using information from these tables, the Capture Throughput Analysis window displays the calculated results of the performance rates of four different tasks.

You can examine the results of all four types of information to assess the throughput performance of the Capture program. You can specify that the results are displayed in absolute or average values.

- Number of rows inserted from log or skipped
- Number of rows pruned from CD table
- Number of transactions committed
- Use of memory

For example, from the Capture Throughput Analysis window, you can review the average weekly performance of the Capture program throughput. To do so, specify the dates and times for both the beginning and the end of a time range, and then specify that the results be displayed as an average of the calculated rates.

## Displaying latency of data processed by the Capture program

Use the Capture Latency window to display the approximate length of time between when data was updated at the source and when it was captured by the Capture program. The elapsed time gives you an indication of the currency of the data in those CD tables over time. This average latency is derived from information the Capture monitor (CAPMON) table, which derives its information from the register (IBMSNAP\_REGISTER) table.

The current Capture latency is calculated using the CURRENT\_TIMESTAMP value in the SYNCHTIME column from the global record in the register (IBMSNAP\_REGISTER) table:

$(\text{CURRENT\_TIMESTAMP}) - (\text{SYNCHTIME})$

*Table 10. Example of values for calculating current Capture latency*

Parameter	Column value
CURRENT_TIMESTAMP	2001-10-20-10:30:25
SYNCHTIME	2001-10-20-10:30:00

For example, using the values in Table 10, the current latency is 25 seconds:

10:30:25 - 10:30:00

The Capture latency changes as time goes by and the history of these changes is stored in the Capture monitor (IBMSNAP\_CAPMON) table. The Replication Center uses the information in the Capture monitor table to calculate average or historical latency. The formula for average latency is similar to the one used for current latency; however, the MONITOR\_TIME value is used instead of the CURRENT\_TIMESTAMP value. The MONITOR\_TIME value is a timestamp indicating when the Capture program inserted a row in the Capture monitor table. You can show the average latency per second, minute,

hour, day, or week. For example, from the Capture Latency window, you can display the average latency for a Capture program per hour, over the duration of the past week.

## **Reviewing Apply program messages**

Use the Apply Report window to check the success of a specific Apply program over a specific period of time by reviewing the data that was inserted into the Apply trail (IBMSNAP\_APPLYTRAIL) table. The IBMSNAP\_APPLYTRAIL table contains data about the execution of subscription sets, and includes the status of the subscription set, error messages, and the number of rows processed.

In the Apply Report window, you can display the following data:

- All subscription sets
- Failed subscription sets
- Successful subscription sets
- Error summary per failed subscription set

For example, from the Apply Report window, you can determine whether the Apply program successfully processed subscription sets during the last week. If there were subscription sets that could not be replicated, you can view the error messages issued by the Apply program for those sets. In addition, you can use the Apply Report window with the Apply Throughput Analysis window. After you use the Apply Report window to find out which sets were successfully replicated, you can use the Apply Throughput Analysis window to see the number of rows replicated and the length of time that replication took.

## **Examining Apply program throughput**

Use the Apply Throughput Analysis window to examine the performance statistics for a specific Apply qualifier. You can filter and group data without writing SQL statements. For example, you can view the number of rows that were inserted into, updated in, deleted from, and reworked in the target tables in the subscription set that was processed by a specific Apply qualifier. You can also view the time the Apply program spent processing subscription sets for a particular Apply qualifier.

## **Displaying the average length of time taken to replicate transactions**

Use the End-to-End Latency window to display an approximate value for the average length of time used to replicate transactions in a particular subscription set. See the Replication Center help for a description of the sequence of events that take place in change-capture replication.

From the End-to-End Latency window, for example, you can view the approximate latency for a subscription set for each Apply cycle during a range of time. You can also divide the time into intervals and display the average latency for each interval.

The Replication Center uses the following formula to calculate the end-to-end latency:

$$(\text{ENDTIME} - \text{LASTRUN}) + (\text{SOURCE\_CONN\_TIME} - \text{SYNCHTIME})$$

where:

- ENDTIME is the time at which the Apply program finishes processing the subscription set.
- LASTRUN is the time at which the Apply program starts processing a subscription set.
- SOURCE\_CONN\_TIME is the time at which the Apply program connects to the Capture control server to fetch data.
- SYNCHTIME is the time of the most current commit of data to the CD tables by the Capture program.

*Table 11. Example of values for calculating end-to-end latency*

Parameter	Column value
ENDTIME	2001-10-20-10:01:00
LASTRUN	2001-10-20-10:00:30
SOURCE_CONN_TIME	2001-10-20-10:00:32
SYNCHTIME	2001-10-20-10:00:00

For example, assume a particular subscription set has the values that are shown in Table 11. Using the previous equation, the average end-to-end latency for this subscription set is 62 seconds:

$$(10:01:00 - 10:00:30) + (10:00:32 - 10:00:00) = 62$$

---

## Setting up automated monitoring of your replication environment

After you set up your replication environment and start the Capture and Apply programs, you want to know that the Capture program is capturing data and that the Apply program is applying it, as you specified. If either program is not working, you want to know when something occurred that caused it to stop. Use the automated Replication Alert Monitor to check the operation and performance of your replication environment.

The Replication Alert Monitor runs on DB2 for UNIX, Windows, or z/OS, and can monitor database servers on those platforms as well as on DB2 for iSeries.

For example, you might want to set up automatic monitoring on a dedicated NT server in your environment, and from that NT server you can run the Replication Alert monitor. The monitor program can connect to all your replication servers and monitor the activity of all the Capture and Apply programs on any platforms in your enterprise. It stores information in monitor control tables on the Monitor control server. A Monitor control server can be on DB2 for UNIX, Windows, or z/OS.

The Replication Alert Monitor can monitor the activity of the Capture and Apply programs after you select the alert conditions that you want to monitor. While the Replication Alert Monitor is running, it checks for those alert conditions. It logs any detected alert conditions. You can set up contacts (consisting of names and e-mail addresses), or groups of contacts, if you want the Replication Alert Monitor to automatically notify those contacts by e-mail when it detects an alert condition. You can also set up the monitor so that it sends e-mail when an operational error occurs.

The Replication Alert Monitor does not monitor triggers associated with non-DB2 relational databases used as sources in a federated database system.

The following sections describe how to use the Replication Alert Monitor for automated monitoring of your replication environment:

- “Creating Monitor control tables”
- “Defining contact information for the Replication Alert Monitor” on page 170
- “Selecting alert conditions for the Replication Alert Monitor” on page 171
- “Scheduling when to start the Replication Alert Monitor” on page 178
- “Starting the Replication Alert Monitor” on page 173
- “Reinitializing the Replication Alert Monitor” on page 179
- “Stopping the Replication Alert Monitor” on page 179

## **Creating Monitor control tables**

To monitor your replication environment with the Replication Alert Monitor, you must first create Monitor control tables on a server. The server on which you create the Monitor control tables is called a Monitor control server. DB2 for UNIX, Windows, z/OS, or iSeries servers can be a Monitor control server. Ensure that the Replication Alert Monitor runs on the Monitor control server.

Before you set up the monitor control tables, you must decide on your monitoring strategy for your replication configuration. You can run a Replication Alert Monitor on each server where you have replication programs running (except on DB2 for iSeries servers), or you can define a centralized Monitor control server. If you use a centralized Monitor control server, it runs remotely from all of your replication programs, it obtains

information by performing remote connects, and it consolidates the information in a central location. Evaluate the needs of your enterprise to determine the appropriate monitoring configuration. For example, the advantage of the centralized approach is its simple monitoring configuration that consolidates data. The centralized approach also has its disadvantages: it takes more time for the monitor to detect and report an alert condition, and the monitor cannot detect problems if it loses connectivity with the remote servers.

**Procedure:**

You can create Monitor control tables using the following method:

**Replication Center**

Use the Create Monitor Control Tables window. See the Replication Center help for details.

You can drop Monitor control tables using the following method:

**Replication Center**

Use the Drop Monitor Control Tables window. See the Replication Center help for details.

**Defining contact information for the Replication Alert Monitor**

Before you can monitor your replication environment with the Replication Alert Monitor, you must define the contact information for the individuals or groups whom you want to notify of an alert condition. When an alert condition occurs, the Replication Alert Monitor uses an e-mail address to send notification to your individual or group contact. The contacts you define can be used by all instances of the Replication Alert Monitor that use the Monitor control server where the contacts are defined. Therefore, if you have multiple Monitor control servers, you must define contacts for each of them.

After you define contacts (by specifying the name and e-mail address of the contact), you can group contacts as necessary. You group contacts by specifying a name for the group (for example, DB2 administrators) and selecting the contacts that you want to be part of that group.

**Procedure:**

To define contact information use the following method:

**Replication Center**

Use the Create Contact window or the Create Contact Group window. See the Replication Center help for details.

To change the information for a defined contact or contact group use the following method:



**Replication Center**

Use the Contact Properties window or the Contact Group Properties window. See the Replication Center help for details.

To copy your contact information to another Monitor control server use the following method:

**Replication Center**

Use the Copy Contacts and Contact Groups window. See the Replication Center help for details.

These contacts are unique to the Replication Center. The Replication Center does not recognize contacts that you created in the Task Center or the Health Center.

**Selecting alert conditions for the Replication Alert Monitor**

You can use the Replication Alert Monitor to track the following information or alert conditions:

**Alert conditions for the status of the Capture or Apply programs**

The Replication Alert Monitor can send an alert notification if a Capture or Apply program terminates.

**Alert conditions for error or warning messages issued by a Capture program or by an Apply program**

The Replication Alert Monitor can send an alert notification if an error or warning message is inserted by the Capture program in the Capture trace (IBMSNAP\_CAPTRACE) table, or by the Apply program in the Apply trace (IBMSNAP\_APPLYTRACE) or Apply trail (IBMSNAP\_APPLYTRAIL) table.

**Alert conditions for thresholds being exceeded**

The Replication Alert Monitor can send an alert notification if any of the following thresholds are exceeded:

- A Capture program is using more memory than you have allowed it.
- The latency for a Capture program exceeds the limit that you have specified.
- The historic latency for a Capture program exceeds the limit that you have specified.
- The end-to-end latency for a transaction exceeds the limit that you have specified.
- An Apply program has reworked more rows in a subscription set than you have allowed it.

### **Alert conditions for miscellaneous events**

The Replication Alert Monitor can send an alert notification if any of the following events occur:

- An Apply program rejected a transaction in an update-anywhere scenario due to conflict detection.
- An Apply program tried to perform a full refresh of a subscription set.
- Processing for a subscription set failed.
- Processing for a subscription set was delayed.
- An Apply program rendered a subscription set inactive.

The Replication Alert Monitor can be used to monitor only Capture and Apply programs whose control tables are at DB2 replication Version 8 architectural level or later. When you select alert conditions, you must provide a Monitor qualifier.

The Replication Alert Monitor monitors the activity of the Capture and Apply programs for all the alert conditions you specify, in the time intervals that you specify when you start the Replication Alert Monitor. You can also change alert conditions after the Replication Alert Monitor starts.

### **Procedure:**

To select alert conditions for the Capture program use one of the following methods:

#### **Replication Center**

Use the Select Alert Conditions for Capture Schemas window, or the Properties window. See Replication Center help for details.

To avoid detecting unnecessary alert conditions, specify thresholds that are compatible with your environment. For example, if Capture is running with a 30 second commit interval, do not specify a 10 second threshold for Capture latency because you will receive an alert most of the time.

To select alert conditions for the Apply program or for subscription sets use one of the following methods:

#### **Replication Center**

Use the Select Alert Conditions for Apply qualifiers or Subscription Sets window, or the Properties window. See Replication Center help for details.

To avoid detecting unnecessary alert conditions, specify thresholds that are compatible with your environment. For example, if you are running the Apply program using time-based scheduling and you schedule the subscription sets

to run every 10 minutes, do not specify an 8 minute threshold for subscriptions delay alert condition because you will receive an alert most of the time. Instead, make sure that you set that threshold to a value that is greater than 10 minutes.

## Starting the Replication Alert Monitor

### Prerequisites:

Before you start the Replication Alert Monitor:

- Ensure that you have created the monitor control tables. See “Creating Monitor control tables” on page 169 for details.
- Ensure that you have created a password file and set up authorization to access control tables for monitoring. Also, ensure that you have bound the Replication Alert Monitor. See Chapter 2, “Setting up for replication” on page 15 for details.
- Ensure that you have selected the alert conditions that you want to monitor. See “Selecting alert conditions for the Replication Alert Monitor” on page 171 for details.

### Procedure:

To start the Replication Alert Monitor use one of the following methods:

#### **Replication Center (UNIX, Windows, z/OS)**

Use the Start monitor window. See the Replication Center help for details.

#### **asnmon system command (UNIX, Windows, z/OS)**

See “asnmon: Starting the Replication Alert Monitor (UNIX, Windows, z/OS)” on page 330 for details.

#### **Windows Services (Windows)**

See Chapter 20, “Using the Windows Service Control Manager to issue system commands (Windows)” on page 459 for details.

#### **MVS console or TSO (z/OS)**

See Chapter 19, “Operating the replication programs (z/OS)” on page 453 for details.

You can start the Replication Alert Monitor to monitor more than one Capture control server or Apply control server at a time. For example, you might consider starting two Replication Alert Monitor programs to distribute the workload among the Monitor control servers or to ensure that a mission-critical application has a dedicated instance of the Replication Alert monitor. For each instance of the Replication Alert Monitor, you must specify a different Monitor qualifier. The Replication Alert Monitor runs in its own processing thread, independently of the Capture and Apply programs.

The Monitor control server and the monitor qualifier are required to start the Replication Alert Monitor. The following sections describe how you can control the behavior of the Replication Alert Monitor when you start it:

- “Specifying how the Replication Alert Monitor runs”
- “Selecting the output for log messages from the Replication Alert Monitor”
- “Specifying prune intervals for data from the Replication Alert Monitor” on page 175
- “Specifying notification criteria for selected alert conditions” on page 176

### **Specifying how the Replication Alert Monitor runs**

Defaults:

**monitor\_interval**=300 seconds (5 minutes)

**runonce**= n

When you start the Replication Alert Monitor, by default, it runs at intervals to monitor any alert conditions that you selected. You can schedule the Replication Alert Monitor to run hourly, at some other time interval, or even just one time. Use the **runonce** parameter or the **monitor\_interval** parameter to modify the behavior of the **asnmon** command. Alternatively, use the Replication Center to specify run times when you start the Replication Alert Monitor.

When you specify **runonce=y**, the Replication Alert Monitor checks one time for all the alert conditions you selected and the **monitor\_interval** parameter is ignored. You can use **runonce** when you include in a batch process the operation of the Replication Alert Monitor. For example, after the Apply program completes, you can use **runonce=y** to determine if any subscription sets failed. Then, if a subscription set did fail, the Replication Alert Monitor sends notification to your contact person or group.

By default, the **monitor\_interval** is 300 seconds (five minutes). The Replication Alert Monitor checks for all the alert conditions for the specific monitor qualifier every 300 seconds. If the Replication Alert Monitor finds an alert condition, it sends notification.

### **Selecting the output for log messages from the Replication Alert Monitor**

Defaults:

**logreuse**= n

**monitor\_path**= directory where **asnmon** command was invoked

When you start the Replication Alert Monitor, it creates a log file in a directory where it stores its work files and appends messages to that log file. By default, the **monitor\_path** directory is where you started the program. You can change the **monitor\_path** only when you start the Replication Alert Monitor, not while it is operating. You can specify a different **monitor\_path** for the Replication Alert Monitor to store its log file.

The name of the log file is *db2instance.monitor\_cntl\_server.monitor\_qual.MON.log*. For example, DB2INST.MONDB1.MONQUAL.MON.log.

By default, the Replication Alert Monitor appends messages to the log file, even after the Replication Alert Monitor is restarted. Keep this default (**logreuse= n**) if you want to maintain a history of messages in the program log file. Ensure that there is enough space in the **monitor\_path** directory for the program log file to grow.

In some situations you might want the Replication Alert Monitor to delete the log and recreate it when it restarts. In these cases, specify the parameter variable **logreuse=y**:

- The log is getting large and you want to clean out the log.
- You do not need the history stored in the log.
- You want start the Replication Alert Monitor from a batch file, and, because of space limitations, you want the log file in a different location.

The **logstdout** parameter is available only if you use the **asnmon** command, it is not available in the Replication Center. By default, the Replication Alert Monitor sends messages to the log file only (**logstdout= n**). If you are troubleshooting or actively monitoring the operation of the Capture or Apply program, you can specify that the Replication Alert Monitor send messages to standard output (**logstdout=y**).

### Specifying prune intervals for data from the Replication Alert Monitor Defaults:

**autoprune=y**

**alert\_prune\_limit=10080** minutes (seven days)

**trace\_limit=10080** minutes (seven days)

Automatic pruning is controlled by the **autoprune** parameter. By default, **autoprune=y**, so the Replication Alert Monitor automatically prunes rows from the IBMSNAP\_ALERTS table that it has already copied into the Monitor control tables.

When the Replication Alert Monitor starts a new monitor cycle, it prunes the Monitor alerts (IBMSNAP\_ALERTS) table if any rows are eligible for pruning. By default, the Replication Alert Monitor prunes the rows that are older than 10080 minutes (seven days). By changing the value for the **alert\_prune\_limit** parameter, you can control how much old data you want to store in the table by specifying how old the data must be before it is pruned from the table.

You might want to reduce the value of **alert\_prune\_limit** parameter if the Replication Alert Monitor is recording many alert conditions or if the storage space is small for the IBMSNAP\_ALERTS table. Alternatively, you might want to maintain a history of all the alert activity. In this case, you can increase the value for **alert\_prune\_limit** parameter.

Also by default, the rows in the Monitor trace (IBMSNAP\_MONTRACE) table and in the Monitor trail (IBMSNAP\_MONTRAIL) table are stored for 10080 minutes (seven days). That is the default of the **trace\_limit** parameter. Any rows older than the value you specify for the **trace\_limit** parameter are pruned.

### **Specifying notification criteria for selected alert conditions**

Defaults:

**max\_notifications\_per\_alert**=3 notifications for a given alert

**max\_notifications\_minutes**=60 minutes

If the Replication Alert monitor detects any alert conditions that you selected, it stores them. You can set up notification parameters to ensure that someone is notified of the alert conditions automatically via e-mail.

The number of notifications you receive also depends on the values you set for the **max\_notifications\_minutes** and **max\_notifications\_per\_alert** parameters.

By default, if an alert condition is met more than once, the Replication Alert Monitor sends a maximum of three notifications for that alert condition in a period of 60 minutes. Use the **max\_notifications\_per\_alert** parameter to specify the maximum number of notifications you want to receive if a particular alert condition occurs within the time period specified by the **max\_notifications\_minutes** parameter. Using these parameters ensures that repeated notification is limited if an alert condition persists for a long period of time before the problem is fixed.

To enable notification, you must also set the **email\_server** parameter. Set the value of this parameter to the address of an e-mail server using the SMTP protocol.

The content of the e-mail notification depends on whether the e-mail address you provided is for a pager. The following examples show the type of information you can expect in each case, for one set of alerts. The e-mail that is sent to non-pager devices shows the time when each alert occurred (at the specific server), the number of times it occurred, and the associated message. The e-mail that is sent to pagers is similar except it contains a summary of the parameters that triggered the alert instead of a complete message. If an alert occurred many times, the timestamp reflects the last time the alert occurred.

The following example shows an example of the notification that is sent.

**Example e-mail notification to non-pager devices:**

```
-----
To:      repladmin@company.com
From:    replmon@server.com
Subject: Monitor: "MONQUAL" Alerts issued

ASN5129I MONITOR "MONQUAL". The Replication Alert Monitor on
server "WSDB" reports an e-mail alert

2002-01-20-10.00.00      1 ASN0552E Capture : "ASN" The program
encountered an SQL error. The server name is "CORP". The SQL
request is "PREPARE". The table name "PROD1.INVOICESCD".
The SQLCODE is "-204". The SQLSTATE is "42704". The SQLERRMC
is "PROD1.INVOICESCD". The SQLERRP is "readCD"

2002-01-20-10.05.00      2 ASN5152W Monitor "MONQUAL". The current
Capture latency exceeds the threshold value. The Capture control
server is "CORP". The schema is "ASN". The Capture
latency is "90" seconds. The threshold is "60" seconds

2002-01-20-10.05.00      4 ASN5154W Monitor "MONQUAL". The memory
used by the Capture program exceeds the threshold value. The
Capture control server is "CORP". The schema is "ASN".
The amount of memory used is "34" megabytes. The threshold is
"30" megabytes.

-----
```

**Example e-mail notification to pagers:**

```
To:      repladmin@company.com
From:    replmon@server.com
Subject: Monitor: "MONQUAL" Alerts issued

MONQUAL - MONDB

2002-01-20-10.00.00 ASN0552E 1 CAPTURE-ERRORS - CORP - ASN
2002-01-20-10.05.00 ASN5152W 2 CAPTURE_CLATENCY - CORP - ASN - 90 - 60
2002-01-20-10.05.00 ASN5154W 4 CAPTURE_MEMORY - CORP - ASN - 34 - 30
```

If the size of the e-mail notification exceeds the limit for the type of e-mail, the notification is sent in multiple e-mail notifications. The size of a regular e-mail notification is limited to 1024 characters. For a pager e-mail address the limit is 250 characters.

Notification is handled by the ASNMAIL exit routine. This exit routine takes the following input:

```
asnmail email_server to-address subject alert-message alert-message
```

where:

- *email\_server* is the address of an e-mail server using the SMTP protocol. This server address is passed from the EMAIL\_SERVER parameter specified in at the start of the asnmon command.
- *to-address* is the e-mail address of the contact to be notified.
- *subject* is the subject in the notification. This is a translated message.
- *alert-message* is a string with the alert message.

Instead of sending alerts through e-mail notification, you can provide your own code to put the alerts in a problem management system. A sample of the asnmail exit is provided, which contains the input parameters. The sample, asnmail.c, is in the sqllib\samples\repl\ directory. For directions on using the sample program, see the prolog of the sample program.

### Specifying notification criteria for operational errors

The Replication Alert monitor stores any errors that occur in the monitoring process, such as the Replication Alert Monitor not being able to connect to the monitor control server. Use the **monitor\_errors** parameter to send notification if an error occurs that is related to the operation of the Replication Alert Monitor. Set the value of this parameter to the e-mail address where you want the notification sent. If you don't specify an e-mail address for this parameter, such errors will be logged but not sent.

This parameter is ignored if the **email\_server** parameter is not specified.

## Scheduling when to start the Replication Alert Monitor

You can schedule the Replication Alert Monitor to start running in the same way you would schedule to start the Capture program or the Apply program. See Chapter 21, "Scheduling replication programs on various operating systems" on page 463 for details.

For information about specifying at start-up whether the Replication Alert Monitor runs continuously or in specific cycles of time, see "Specifying how the Replication Alert Monitor runs" on page 174.



## Reinitializing the Replication Alert Monitor

### Prerequisites:

The Replication Alert Monitor with the specific monitor qualifier must be started.

You can specify if you want the Replication Alert Monitor to obtain values for the contacts, alert conditions, and the Replication Alert Monitor parameters that you changed while the Replication Alert Monitor was operating. For example, reinitialize the Monitor if you add a new e-mail address for a contact while the Replication Alert Monitor is running.

### Procedure:

To reinitialize the Replication Alert Monitor use one of the following methods:

#### Replication Center (UNIX, Windows, z/OS)

In the Replication Center, right-click on the Monitor qualifier and select **Reinitialize Monitor**. See the Replication Center help for details.

#### asnmcmd system command (Windows, UNIX, z/OS)

See “asnmcmd: Operating the Replication Alert Monitor (UNIX, Windows, z/OS)” on page 329 for details about the **asnmcmd** system command, **reinit** parameter.

## Stopping the Replication Alert Monitor

### Prerequisites:

The Replication Alert Monitor with the specific Monitor qualifier must be started.

### Procedure:

To stop the Replication Alert Monitor use one of the following methods:

#### Replication Center (UNIX, Windows, z/OS)

In the operations tree, right click the Monitor qualifier and select **Stop Monitor**. See the Replication Center help for details.

#### asnmcmd system command (Windows, UNIX, z/OS)

See “asnmcmd: Operating the Replication Alert Monitor (UNIX, Windows, z/OS)” on page 329 for details about the **asnmcmd** system command, **stop** parameter.

### Windows Services (Windows)

See Chapter 20, “Using the Windows Service Control Manager to issue system commands (Windows)” on page 459 for details.

### MVS console or TSO (z/OS)

See Chapter 19, “Operating the replication programs (z/OS)” on page 453 for details.

---

## Monitoring the progress of the Capture program (OS/400)

If the Capture program has terminated, you can inspect the restart table (IBMSNAP\_RESTART) to determine how much progress the Capture program made. There is one row for each journal used by the source tables. The LOGMARKER column provides the timestamp of the last journal entry processed successfully. The SEQNBR column provides the journal entry sequence number of that entry.

If the Capture program is still running, you can determine its progress using the following steps:

1. For each source table being captured, open its CD table.
2. In the last row of the CD table, note the hex value in the COMMITSEQ column.
3. Look in the Unit-of-work (IBMSNAP\_UOW) table for a row with the same COMMITSEQ value. If no matching COMMITSEQ exists in the IBMSNAP\_UOW table, repeat the same process with the second-to-last row in the CD table. Proceed backward through the CD table until you find a match.
4. When you find a matching COMMITSEQ, note the value in the LOGMARKER column of the UOW row. This is the timestamp of the processed journal entry. All changes to the source table up to that time are ready to be applied.
5. Use the Display Journal (**DSPJRN**) system command to determine how many journal entries remain to be processed by the Capture program. Direct the output to an output file (or to a printer for a printed report), as shown in the following example:

```
DSPJRN FILE(JRNLIB/DJRN1)
      RCVRNG(*CURCHAIN)
      FROMTIME(timestamp)
      TOTIME(*LAST)
      JRNCDE(J F R C)
      OUTPUT(*OUTFILE)
      ENDTALEN(1) OUTFILE(library/outfile)
```

where *timestamp* is the timestamp that you identified in 4.

The number of records in the output file is the approximate number of journal entries that remain to be processed by the Capture program.



---

## Chapter 12. Making changes to your replication environment

This chapter covers the issues that you will need to consider when you make day-to-day changes to your replication environment.

This chapter contains the following sections:

- “Registering new objects”
- “Changing registration attributes for registered objects” on page 184
- “Adding columns to source tables” on page 185
- “Stop capturing changes for registered objects” on page 188
- “Reactivating registrations” on page 189
- “Removing registrations” on page 190
- “Changing Capture schemas” on page 191
- “Creating new subscription sets” on page 194
- “Adding new subscription-set members to existing subscription sets (UNIX, Windows, z/OS)” on page 195
- “Changing attributes of subscription sets” on page 198
- “Changing subscription set names” on page 199
- “Splitting a subscription set” on page 201
- “Merging subscription sets” on page 206
- “Changing Apply qualifiers of subscription sets” on page 209
- “Deactivating subscription sets” on page 212
- “Removing subscription sets” on page 213
- “Coordinating replication events with database application events” on page 214
- “Promoting your replication configuration to another system” on page 221

---

### Registering new objects

You can register a new table, view, or nickname in your replication environment at any time. You do not need to reinitialize the Capture program.

#### **Procedure:**

Use one of the following methods to register an object:

### **Replication Center**

Use the Register Tables, Register Views, or Register Nicknames window. See the Replication Center help for details.

### **ADDDPRREG system command (OS/400)**

See “ADDDPRREG: Adding a DPR registration (OS/400)” on page 349 for parameter descriptions and command syntax.

A new registered object is automatically initialized by the Capture program the first time that the Apply program processes a subscription set that refers to that object. The Apply program signals the Capture program to begin capturing changes for this new object. See Chapter 3, “Registering tables and views as replication sources” on page 37 for more information about registering objects.

---

## **Changing registration attributes for registered objects**

You can change the registration attributes of existing registered objects at any time. These registration attributes include:

- CHGONLY
- CONFLICT\_LEVEL
- RECAPTURE
- DISABLE\_REFRESH
- CHG\_UPD\_TO\_DEL\_INS
- STOP\_ON\_ERROR
- BEFORE\_IMG\_PREFIX

**Note:** You can update the before-image prefix value only if this value is null.

### **Procedure:**

1. Use the following method to change the attributes:

#### **Replication Center**

From the Registered Tables folder, right-click the registered table in the contents pane and select Properties. See the Replication Center help for details.

2. After you change the attributes, you must reinitialize the Capture program in order for it to recognize the changes. Reinitialize the Capture program by using one of the following methods:

#### **Replication Center**

From the Capture Control Servers folder, right-click the Capture control server in the contents pane and select Reinitialize Capture. See the Replication Center help for details.

**asnccmd system command (Windows, UNIX, z/OS)**

Use the **reinit** parameter. See “asnccmd: Operating Capture (UNIX, Windows, z/OS)” on page 322 for parameter descriptions and command syntax.

**INZDPRCAP system command (OS/400)**

See “INZDPRCAP: Reinitializing DPR Capture (OS/400)” on page 412 for parameter descriptions and command syntax.

---

**Adding columns to source tables**

If you need to add columns to a registered source table, first consider how DB2 replication uses this table. If you need to replicate the new columns in this source table, you must ensure that the existing Capture and Apply programs recognize the new columns and continue processing without interruption. You might need to perform special processing steps depending on whether or not you want to replicate the data in the new columns.

**Not replicated**

If you do not want to replicate the data in the new columns, you do not need to perform any special processing steps. The Capture program immediately recognizes the changes and continues running.

**Replicated**

If you want to replicate the data in these new columns, follow these steps to ensure that the new column data is captured and that the Capture and Apply programs continue to run without errors.

**Prerequisite:**

Before using this procedure, familiarize yourself with the structures of your source, change-data (CD), and target tables and with the registrations and subscription sets defined on your system.

**Restrictions:**

Do not use these steps if you are adding columns to an iSeries table that uses a relative record number (RRN) as the primary key. The RRN must be the last column in the CD table. When adding columns to an iSeries table with an RRN, remove the registration, add the column to the source table, and then add this table again as a new registration specifying that the RRN will be captured. See “RMVDPRREG: Removing a DPR registration (OS/400)” on page 420 and “ADDDPRREG: Adding a DPR registration (OS/400)” on page 349 for more information about removing and adding iSeries registrations.

You cannot use these steps to add columns to registered sources on non-DB2 relational databases. A registration for a non-DB2 relational

source includes a set of triggers used for capturing changes. You cannot alter these triggers. Therefore, if you need to add new columns to this source table and need to replicate the data in these columns, you must drop and re-create the existing registered source.

**Procedure:**

1. Quiesce all activity against the source table that you want to alter.
2. Use one of the following methods to stop the Capture program:

**Replication Center**

Use the Stop Capture window. See the Replication Center help for details.

**asncmd system command (Windows, UNIX, z/OS)**

Use the **stop** parameter. See “asncmd: Operating Capture (UNIX, Windows, z/OS)” on page 322 for parameter descriptions and command syntax.

**ENDDPRCAP system command (OS/400)**

See “ENDDPRCAP: Stopping Capture (OS/400)” on page 400 for parameter descriptions and command syntax.

**Tip:** If you need to keep the Capture program active during this procedure, insert a USER signal in the signal (IBMSNAP\_SIGNAL) table after stopping activity against the source table. Wait for the Capture program to process the USER signal.

After the Capture program processes the USER signal, the Capture program has no more activity to process against the associated CD table and no longer requires access to this CD table.

3. Use the following method to deactivate all subscription sets that subscribe to this source table:

**Replication Center**

From the Subscription Sets folder, right-click the active subscription sets in the contents pane and select Deactivate. See the Replication Center help for details.

**Note:** If you do not want to deactivate the subscription sets during this process, verify that no Apply programs associated with these subscriptions sets will be running against this source table when you are adding the new columns. Alternatively,



ensure that these Apply programs have processed data up to the signal log sequence number (LSN) that is associated with the prior USER signal.

The methods in this step ensure exclusive access to the CD table so that you can alter the table.

4. Use SQL to submit an ALTER TABLE ADD statement to add the new columns to the source table.
5. Use the following method to add the new columns to the CD table:

#### **Replication Center**

From the Registered Tables folder, right-click the registered table in the contents pane and select Properties. See the Replication Center help for details.

The Capture program automatically reinitializes the registration and captures the changes to these new columns when the Capture program first reads log data with the new columns.

6. Use SQL to submit an ALTER TABLE ADD statement to add the new columns to the target table.
7. Use the following method to deactivate any associated subscription sets that you did not already deactivate in step 3:

#### **Replication Center**

From the Subscription Sets folder, right-click the active subscription sets in the contents pane and select Deactivate. See the Replication Center help for details.

If absolutely necessary, you can now resume activity against this source table. However, because the associated subscriptions sets have not yet been changed, you must keep these subscription sets deactivated so that you do not lose any changes made to these new columns.

8. Use the following method to add the new columns to the associated subscription-set members:

#### **Replication Center**

Use the Add Column to Target Table window. See the Replication Center help for details.

9. **For UNIX, Windows, z/OS:** If you are running the Apply program with **opt4one** set to y, stop and then restart the Apply program.
10. Use the following method to reactivate the subscription sets:

## Replication Center

From the Subscription Sets folder, right-click the deactivated subscription sets in the contents pane and select Activate. See the Replication Center help for details.

---

## Stop capturing changes for registered objects

You should deactivate a registered object before you delete it to ensure that the Capture programs finish any necessary processing of the object. Also, you can deactivate a registered object if you want to stop capturing changes for this object temporarily but need to keep your Capture programs running for other registered objects.

The Capture program stops capturing changes for the source objects that have been deactivated; however, the change-data (CD) tables, registration attributes, and subscription sets that are associated with these source objects remain on the system.

Before you deactivate a registered object, you should deactivate all of the subscriptions sets that are associated with this registered object. This ensures that your Apply programs will not interfere with the deactivation process by automatically reactivating the object before you delete it or before you are ready to reactivate it.

All subscription sets that are associated with the registered object are affected when the object is deactivated and when DB2 replication stops capturing changes for that object. If you want to continue running these subscription sets, you must remove the subscription-set members that use this registered object as a source from the deactivated subscription sets.

### Restrictions:

You can deactivate only DB2 registered objects that are defined as Capture program sources.

You cannot deactivate non-DB2 relational database objects that are used by Capture triggers.

### Procedure:

To deactivate a registered object:

1. Deactivate all associated subscription sets using the following method:

#### Replication Center

From the Subscription Sets folder, right-click the active

subscription sets in the contents pane and select Deactivate. See the Replication Center help for details.

See “Deactivating subscription sets” on page 212 for more information.

2. Deactivate the registered object using one of the following methods:

**Replication Center**

From the Registered Tables folder, right-click the registered table in the contents pane and select Stop Capturing Changes. See the Replication Center help for details.

**CAPSTOP signal**

Manually insert a CAPSTOP signal in the signal (IBMSNAP\_SIGNAL) table. See “*schema.IBMSNAP\_SIGNAL*” on page 507 for more information.

---

## Reactivating registrations

If you temporarily deactivate your registration and associated subscription sets and then want to reactivate the registration to begin recapturing data, you would simply reactivate these subscription sets through the Replication Center. The Capture program reactivates the registration after the Apply program sends a CAPSTART signal.

If, however, the Capture program deactivates a registration because of an unexpected error, you must take special action to reactivate the registration. Unexpected errors cause the Capture program to set the value of the STATE column to S (Stopped) in the register (IBMSNAP\_REGISTER) table if the STOP\_ON\_ERROR column value for this registration is set to N. This STATE column value indicates that the Capture program stopped processing this registration and that the registration must be repaired. The Apply program does not issue a CAPSTART signal for any registration that is in a stopped state.

Use the following procedure to correct these unexpected errors and to make the registration eligible for reactivation.

**Prerequisites:**

Read the error messages that were generated by the Capture program regarding this deactivated registration.

Familiarize yourself with the structure of the DB2 replication Capture control tables and with the Capture programs running on your system.

**Procedure:**

1. Change your registration by using the information contained in the error messages.
2. From the Capture control server, run the following SQL script to reset the STATE column in the IBMSNAP\_REGISTER table:

```
UPDATE Schema.IBMSNAP_REGISTER
SET STATE = 'I'
WHERE
    SOURCE_OWNER      = 'SrcSchema' AND
    SOURCE_TABLE      = 'SrcTbl'    AND
    SOURCE_VIEW_QUAL  = SrcVwQual  AND
    STATE              = 'S';
```

where *Schema* is the name of the Capture schema, *SrcSchema* is the registered source table schema, *SrcTbl* is the name of the registered source table, and *SrcVwQual* is the source-view qualifier for this source table.

After the STATE column is set to I (Inactive), the Capture program is ready to begin capturing data as soon as a CAPSTART signal is received, usually from the Apply program.

**Example:** Suppose that the source table for an active registration was inadvertently altered to DATA CAPTURE NONE (and should be DATA CAPTURE CHANGES). Also, suppose that this registration was defined with STOP\_ON\_ERROR = 'N', which specifies that the Capture program will not stop when it encounters errors. At the next restart or reinitialization of the Capture program, the Capture program will recognize this incorrect condition of the source table and will set the STATE column to S (Stopped) in the register (IBMSNAP\_REGISTER) table for this registration. You will receive an error message when the Apply program tries to process the corresponding subscription set, because the registration will be in a stopped state. You must:

- Correct the setting of the source table through SQL by submitting an ALTER TABLE statement that resets the table option to DATA CAPTURE CHANGES.
- Manually reset the registration from a stopped state to an inactive state, using the above SQL script.

The Apply program will then perform a full refresh of the entire subscription set.

---

## Removing registrations

If you remove a registration, DB2 replication removes the registration of the object, drops the associated change-data (CD) or consistent-change data (CCD) tables, and drops the CCD object nickname and any Capture triggers for non-DB2 relational database sources. The actual source table or view remains in the database.

**Prerequisite:**

Deactivate the source object first to ensure that the Capture program finishes any current processing of this object.

**Important:** Deactivation is an asynchronous process. Be sure that the deactivation process finishes before you remove the object.

**Procedure:**

Use one of the following methods to remove a registration for a source table or view:

**Replication Center**

Use the Delete Registered Tables or Delete Registered Views window. See the Replication Center help for details.

**RMVDPRREG system command (OS/400)**

See “RMVDPRREG: Removing a DPR registration (OS/400)” on page 420 for parameter descriptions and command syntax.

---

**Changing Capture schemas**

You can use the following procedure to change an existing Capture schema.

**Prerequisites:**

Before running the following SQL statements, familiarize yourself with your DB2 replication control tables and with the subscription sets that are defined on your system.

**For UNIX, Windows, z/OS:** If you set up monitoring definitions or started Replication Alert Monitor programs under the Capture schema that you are going to change, drop these monitoring definitions. After you change the Capture schema, re-create the monitoring definitions with the new Capture schema name through the Replication Center. Then, you can reinitialize the associated Replication Alert Monitor programs using the **asnmcmd** system command with the **reinit** parameter. Alternatively, you can stop the Replication Alert Monitor programs using the **asnmcmd** system command with the **stop** parameter and then restart the programs using the **asnmon** system command.

Determine the new Capture schema name. See Chapter 16, “Naming rules for replication objects” on page 301 for more information.

Verify that your Capture control server and all of the Apply control servers that are associated with this Capture control server have been migrated to Version 8 before you use this procedure.

**Restriction:**

You should not use this procedure if your source server is a non-DB2 relational database.

**Procedure:**

1. Use one of the following methods to create control tables for a new Capture schema:

**Replication Center (UNIX, Windows, z/OS)**

Use the Create Replication Control Tables notebook. See the Replication Center help for details.

**CRTDPRTBL system command (OS/400)**

See “CRTDPRTBL: Creating the replication control tables (OS/400)” on page 396 for parameter descriptions and command syntax.

2. Use one of the following methods to stop the Capture program that is using the existing Capture schema. (Skip this step if you do not have a Capture program running.):

**Replication Center**

Use the Stop Capture window. See the Replication Center help for details.

**asnccmd system command (UNIX, Windows, z/OS)**

Use the **stop** parameter. See “asnccmd: Operating Capture (UNIX, Windows, z/OS)” on page 322 for parameter descriptions and command syntax.

**ENDDPRCAP system command (OS/400)**

See “ENDDPRCAP: Stopping Capture (OS/400)” on page 400 for parameter descriptions and command syntax.

3. Use the following method to deactivate all associated subscription sets:

**Replication Center**

From the Subscription Sets folder, right-click the active subscription sets in the contents pane and select Deactivate. See the Replication Center help for details.

4. From the Apply control server, run the following SQL statement to change the Capture schema names for the associated subscription sets with source tables that belong to this Capture schema:

```
UPDATE ASN.IBMSNAP_SUBS_SET
SET CAPTURE_SCHEMA = 'NewSchema'
WHERE
    CAPTURE_SCHEMA = 'ExistingSchema';
```

where *NewSchema* is the new Capture schema name, and *ExistingSchema* is the name of the Capture schema that you are changing.

5. If you created subscription sets with target tables (for example, CCD or replica type tables) that are registered in this Capture schema, run the following SQL statement from the Apply control server to change the target schema name of these subscription sets:

```
UPDATE ASN.IBMSNAP_SUBS_SET
SET TGT_CAPTURE_SCHEMA = 'NewSchema'
WHERE
    TGT_CAPTURE_SCHEMA = 'ExistingSchema';
```

where *NewSchema* is the new Capture schema name, and *ExistingSchema* is the name of the Capture schema that you are changing.

6. From the Capture control server, run an SQL statement to copy the active information from each existing Capture control table to each new corresponding Capture control table that you created in step 1. For example, to copy the active information to the new register (IBMSNAP\_REGISTER) table:

```
INSERT INTO NewSchema.IBMSNAP_REGISTER
SELECT * FROM
    ExistingSchema.IBMSNAP_REGISTER;
```

where *NewSchema* is the new Capture schema name, and *ExistingSchema* is the name of the Capture schema that you are changing.

Repeat this step for each existing Capture control table, including some or all of the following tables:

- IBMSNAP\_CAPMON
- IBMSNAP\_CAPPARMS
- IBMSNAP\_CAPTRACE
- IBMSNAP\_PRUNCNTL
- IBMSNAP\_PRUNE\_SET
- IBMSNAP\_REG\_EXT (OS/400 only)
- IBMSNAP\_REGISTER
- IBMSNAP\_RESTART
- IBMSNAP\_SIGNAL
- IBMSNAP\_UOW

(You do not need to repeat this step for the IBMSNAP\_CAPENQ [on UNIX, Windows, z/OS] or the IBMSNAP\_PRUNE\_LOCK control table, because there are no rows in these tables.)

Do *not* change the CD tables.

7. Use the following method to drop the existing schema and its associated Capture control tables:

**Replication Center**

From the Capture Control Servers folder, right-click the database for which you want to remove Capture control tables and select Drop Capture Control Tables. See the Replication Center help for details.

8. Use one of the following methods to restart the Capture program with the new schema name:

**Replication Center**

Use the Start Capture window. See the Replication Center help for details.

**asncap system command (UNIX, Windows, z/OS)**

Use the **capture\_schema=NewSchema** and **startmode=warmsi** or **warmns** parameter options. See “asncap: Starting Capture (UNIX, Windows, z/OS)” on page 316 for parameter descriptions and command syntax.

**STRDPRCAP system command (OS/400)**

Use the **RESTART(\*YES)** parameter. See “STRDPRCAP: Starting Capture (OS/400)” on page 436 for parameter descriptions and command syntax.

9. Reactivate the associated subscription sets using the following method:

**Replication Center**

From the Subscription Sets folder, right-click the deactivated subscription sets in the contents pane and select Activate. See the Replication Center help for details.

---

## Creating new subscription sets

You can create new subscription sets and add new subscription-set members to sets at any time for an existing registered object.

This procedure addresses the addition of a new subscription set, with or without subscription-set members.

**Prerequisites:**



Before you create a new subscription set, register the tables or views that you want to use as sources.

**Procedure:**

Use one of the following methods to create a new subscription set:

**Replication Center**

Use the Create Subscription Set notebook. See the Replication Center help for details.

**ADDDPRSUB system command (OS/400)**

See “ADDDPRSUB: Adding a DPR subscription set (OS/400)” on page 359 for parameter descriptions and command syntax.

See Chapter 4, “Subscribing to sources” on page 63 for additional information.

**Important:** If the corresponding Apply program is active, do not activate the new subscription set until the subscription set is fully defined.

---

## **Adding new subscription-set members to existing subscription sets (UNIX, Windows, z/OS)**

If you add a subscription-set member to a subscription set, DB2 replication forces a full refresh of all the members in the subscription set. Use the following procedure to add a new subscription-set member to an existing subscription set without forcing a full refresh of the entire set.

**Prerequisite:**

Before running these SQL statements, familiarize yourself with the structure of the DB2 replication control tables and with the subscription sets defined on your system.

**Procedure:**

1. Use the following method to deactivate the subscription set:

**Replication Center**

From the Subscription Sets folder, right-click the active subscription set in the contents pane and select Deactivate. See the Replication Center help for details.

2. From the Capture control server, run the following SQL statement to synchronize the synchpoint and synctime values in the pruning control (IBMSNAP\_PRUNCNTL) table with the corresponding values in the prune set (IBMSNAP\_PRUNE\_SET) table:

```

UPDATE Schema.IBMSNAP_PRUNCNTL
SET SYNCHPOINT =
    (SELECT SYNCHPOINT FROM Schema.IBMSNAP_PRUNE_SET
     WHERE APPLY_QUAL = 'ApplyQual' AND
           SET_NAME = 'Name' AND
           TARGET_SERVER = 'Target_Server'),
SYNCHTIME =
    (SELECT SYNCHTIME FROM Schema.IBMSNAP_PRUNE_SET
     WHERE APPLY_QUAL = 'ApplyQual' AND
           SET_NAME = 'Name' AND
           TARGET_SERVER = 'Target_Server')
WHERE
    APPLY_QUAL = 'ApplyQual' AND
    SET_NAME = 'Name' AND
    TARGET_SERVER = 'Target_Server';

```

where *Schema* is the name of the Capture schema, *ApplyQual* is the Apply qualifier, *Name* is the name of the subscription set that you want to add a member to, and *Target\_Server* is the database location of the target tables.

By running this SQL statement, you ensure that the Apply program starts from the correct point in the change-data (CD) tables for each subscription-set member.

3. Use the following method to add the new subscription-set member to this subscription set:

### Replication Center

Use the Add Member to Subscription Set notebook. See the Replication Center help for details.

Keep this subscription set deactivated.

4. From the Capture control server, insert a CAPSTART signal for this new subscription-set member.
  - a. Find the value of the MAP\_ID for this new subscription-set member by running the following SQL SELECT statement:

```

SELECT MAP_ID FROM Schema.IBMSNAP_PRUNCNTL
WHERE
    SOURCE_OWNER = 'SrcSchema' AND
    SOURCE_TABLE = 'SrcTbl' AND
    SOURCE_VIEW_QUAL = 'SrcVwQual' AND
    APPLY_QUAL = 'ApplyQual' AND
    SET_NAME = 'Name' AND
    TARGET_SERVER = 'Target_Server' AND
    TARGET_OWNER = 'TgtSchema' AND
    TARGET_TABLE = 'TgtTbl'
WITH UR;

```

where *Schema* is the name of the Capture schema, *SrcSchema* is the source table schema, *SrcTbl* is the source table name, *SrcVwQual* is the source-view qualifier for this source table, *ApplyQual* is the Apply

qualifier, *Name* is the name of subscription set to which you are adding a member, *Target\_Server* is the database location of the target tables, *TgtSchema* is the schema of the target table, and *TgtTbl* is the target table name.

- b. Take the MAP\_ID value and run the following SQL statement to insert a row into the signal (IBMSNAP\_SIGNAL) table:

```
INSERT INTO Schema.IBMSNAP_SIGNAL
(SIGNAL_TIME,
 SIGNAL_TYPE,
 SIGNAL_SUBTYPE,
 SIGNAL_INPUT_IN,
 SIGNAL_STATE,
 SIGNAL_LSN)
VALUES (CURRENT_TIMESTAMP,
 'CMD',
 'CAPSTART',
 'Mapid',
 'P',
 NULL);
```

where *Schema* is the name of the Capture schema, and *Mapid* is the value of the MAP\_ID that you obtained from the preceding SQL SELECT statement.

5. Ensure that the Capture program picked up the CAPSTART signal.
  - a. Run the following SQL statement from the Capture control server to display the synchpoint value in the IBMSNAP\_PRUNCNTL table:

```
SELECT SYNCHPOINT FROM Schema.IBMSNAP_PRUNCNTL
WHERE
  MAP_ID = 'Mapid'
WITH UR;
```

where *Schema* is the name of the Capture schema and *Mapid* is the value of the MAP\_ID that you obtained from the SQL SELECT statement in step 4a.

- b. Check the value of the synchpoint.
 

If the value is null, repeat steps 5a and 5b until the Capture program recognizes the signal, and the synchpoint value is not null.
6. Using your choice of utility programs, unload the source table data. Then, load this data into the target table.
7. If you are running the Apply program with **opt4one** set to y, stop and then restart the Apply program.
8. From the Apply control server, run the following script to update the subscription sets (IBMSNAP\_SUBS\_SET) table, making the Apply program run immediately for this subscription set and preventing a full refresh:

```

UPDATE ASN.IBMSNAP_SUBS_SET
  SET SYNCHPOINT      = NULL,
      SYNCHTIME       = CURRENT_TIMESTAMP,
      LASTSUCCESS     = CURRENT_TIMESTAMP,
      LASTRUN         = CURRENT_TIMESTAMP - X MINUTES,
      STATUS          = 0,
      ACTIVATE        = 1

WHERE
  APPLY_QUAL          = 'ApplyQual'      AND
  SET_NAME            = 'Name'           AND
  WHOS_ON_FIRST      = 'Val';

```

where *X* is a number that is greater than the `sleep_minutes` setting, *ApplyQual* is the Apply qualifier, *Name* is the name of subscription set to which you added a member, and *Val* is either F or S.

This subscription set is now active, and the Apply program resumes processing the entire subscription set.

---

## Changing attributes of subscription sets

You might need to change an attribute of an existing subscription set. Attributes that you might need to change include:

- Schedules for applying updates (time-based replication or event-based replication)
- Subscription statements
- WHERE clause predicates of subscription-set members
- Commit count
- Data blocking value (`MAX_SYNCH_MINUTES`)

### Procedure:

To change an attribute of a subscription set, perform the following steps through the Replication Center:

1. Deactivate the subscription set.
2. Change the subscription set and any subscription-set members.
3. Reactivate the subscription set.

By first deactivating the subscription set, you keep the Apply program up and running but prevent the Apply program from processing this subscription set while you enter your changes. The Apply program recognizes your subscription set changes during the next Apply cycle after you reactivate the subscription set.

**Note:** If you set the **opt4one** Apply program parameter to y, your changes are not recognized unless you stop and then restart the Apply program (UNIX, Windows, z/OS).

---

## Changing subscription set names

Use the following procedure to change the name of a subscription set without having to drop and then re-create the subscription set and all of its members.

### Prerequisites:

Before running these SQL statements, familiarize yourself with the structure of the DB2 replication control tables and with the subscription sets defined on your system.

**For UNIX, Windows, z/OS:** If you set up monitoring definitions or started Replication Alert Monitor programs to detect alert conditions for the subscription set that you are going to change, drop these monitoring definitions. After you change the subscription-set name, re-create the monitoring definitions through the Replication Center. Then, you can reinitialize the associated Replication Alert Monitor programs using the **asnmcmd** system command with the **reinit** parameter. Alternatively, you can stop the Replication Alert Monitor programs using the **asnmcmd** system command with the **stop** parameter and then restart the programs using the **asnmon** system command.

Determine the new subscription set name that you want to use.

### Procedure:

1. Use the following method to deactivate the subscription set that you want to change:

#### Replication Center

From the Subscription Sets folder, right-click the active subscription set in the contents pane and select Deactivate. See the Replication Center help for details.

2. From the Apply control server, run the following SQL statements to change the name of the subscription set in the subscription sets (IBMSNAP\_SUBS\_SET), subscription members (IBMSNAP\_SUBS\_MEMBR), and subscription columns (IBMSNAP\_SUBS\_COLS) tables:

```
UPDATE ASN.IBMSNAP_SUBS_SET
   SET SET_NAME      = 'NewSetName'
WHERE
   APPLY_QUAL      = 'ApplyQual'      AND
   SET_NAME        = 'ExistSetName'   AND
   WHOS_ON_FIRST   = 'Val';
```

```

UPDATE ASN.IBMSNAP_SUBS_MEMBR
  SET SET_NAME      = 'NewSetName'
  WHERE
    APPLY_QUAL      = 'ApplyQual'      AND
    SET_NAME         = 'ExistSetName'   AND
    WHOS_ON_FIRST   = 'Val';

UPDATE ASN.IBMSNAP_SUBS_COLS
  SET SET_NAME      = 'NewSetName'
  WHERE
    APPLY_QUAL      = 'ApplyQual'      AND
    SET_NAME         = 'ExistSetName'   AND
    WHOS_ON_FIRST   = 'Val';

```

where *NewSetName* is the new subscription set name, *ApplyQual* is the Apply qualifier, *ExistSetName* is the existing name of the subscription set, and *Val* is either F or S.

3. If this subscription set uses before or after SQL statements or procedure calls, run the following SQL script from the Apply control server to change the subscription set name in the subscription statements (IBMSNAP\_SUBS\_STMTS) table:

```

UPDATE ASN.IBMSNAP_SUBS_STMTS
  SET SET_NAME      = 'NewSetName'
  WHERE
    APPLY_QUAL      = 'ApplyQual'      AND
    SET_NAME         = 'ExistSetName'   AND
    WHOS_ON_FIRST   = 'Val';

```

where *NewSetName* is the new subscription set name, *ApplyQual* is the Apply qualifier, *ExistSetName* is the existing name of the subscription set, and *Val* is either F or S.

4. From the Capture control server, run the following SQL statements to change the subscription set name in the prune set (IBMSNAP\_PRUNE\_SET) and pruning control (IBMSNAP\_PRUNCNTL) tables:

```

UPDATE Schema.IBMSNAP_PRUNE_SET
  SET SET_NAME      = 'NewSetName'
  WHERE
    APPLY_QUAL      = 'ApplyQual'      AND
    SET_NAME         = 'ExistSetName'   AND
    TARGET_SERVER    = 'Target_Server';

UPDATE Schema.IBMSNAP_PRUNCNTL
  SET SET_NAME      = 'NewSetName'
  WHERE
    APPLY_QUAL      = 'ApplyQual'      AND
    SET_NAME         = 'ExistSetName'   AND
    TARGET_SERVER    = 'Target_Server';

```

where *Schema* is the name of the Capture schema, *NewSetName* is the new subscription set name, *ApplyQual* is the Apply qualifier, *ExistSetName* is the existing name of the subscription set, and *Target\_Server* is the database location of the target tables.

5. **For UNIX, Windows, z/OS:** If you are running the Apply program with **opt4one** set to **y**, stop and then restart the Apply program.
6. Reactivate the subscription set using the following method:

#### **Replication Center**

From the Subscription Sets folder, right-click the deactivated subscription set in the contents pane and select **Activate**. See the Replication Center help for details.

---

## **Splitting a subscription set**

You can use the following procedure to split a subscription set into two or more subscription sets without having to remove and re-create subscription set information.

### **Prerequisites:**

Before running these SQL statements, familiarize yourself with the structure of the DB2 replication control tables and with the subscription sets defined on your system.

Identify the subscription-set members of the subscription set that you want to split, and determine the source and target tables associated with these subscription-set members.

Identify the Capture control server, target server, and Apply control server of the subscription set that you want to split. You must use these Capture control server, target server, and Apply control server locations for the new subscription set that you want to create using this procedure.

**For UNIX, Windows, z/OS:** If you set up monitoring definitions or started Replication Alert Monitor programs to detect alert conditions for the subscription set that you are going to split, drop these monitoring definitions. After you split the subscription set, re-create the monitoring definitions through the Replication Center. Then, you can reinitialize the associated Replication Alert Monitor programs using the **asnmcmd** system command with the **reinit** parameter. Alternatively, you can stop the Replication Alert Monitor programs using the **asnmcmd** system command with the **stop** parameter and then restart the programs using the **asnmmon** system command.

### **Procedure:**

1. Use the following method to deactivate the subscription set that you want to split:

#### Replication Center

From the Subscription Sets folder, right-click the active subscription set in the contents pane and select Deactivate. See the Replication Center help for details.

2. Use one of the following methods to create a new subscription set:

#### Replication Center

Use the Create Subscription Set notebook. See the Replication Center help for details.

#### ADDDPRSUB system command (OS/400)

Use the **SRCTBL(\*NONE)**, **TGTTBL(\*NONE)**, and **ACTIVATE(\*NO)** parameter options. See “ADDDPRSUB: Adding a DPR subscription set (OS/400)” on page 359 for parameter descriptions and command syntax.

Both of these methods create a new row in the subscription sets (IBMSNAP\_SUBS\_SET) table.

Leave this new subscription set inactive.

3. From the Apply control server, run the following SQL statement to copy information from the existing subscription set into the new subscription set row in the IBMSNAP\_SUBS\_SET table:

```
UPDATE ASN.IBMSNAP_SUBS_SET
  SET STATUS =
    (SELECT STATUS FROM ASN.IBMSNAP_SUBS_SET B
     WHERE APPLY_QUAL = 'ApplyQual' AND
           SET_NAME = 'ExistName' AND
           WHOS_ON_FIRST = 'Val'),
  LASTRUN =
    (SELECT LASTRUN FROM ASN.IBMSNAP_SUBS_SET B
     WHERE APPLY_QUAL = 'ApplyQual' AND
           SET_NAME = 'ExistName' AND
           WHOS_ON_FIRST = 'Val'),
  SYNCHPOINT =
    (SELECT SYNCHPOINT FROM ASN.IBMSNAP_SUBS_SET B
     WHERE APPLY_QUAL = 'ApplyQual' AND
           SET_NAME = 'ExistName' AND
           WHOS_ON_FIRST = 'Val'),
  SYNCHTIME =
    (SELECT SYNCHTIME FROM ASN.IBMSNAP_SUBS_SET B
     WHERE APPLY_QUAL = 'ApplyQual' AND
           SET_NAME = 'ExistName' AND
           WHOS_ON_FIRST = 'Val'),
  LASTSUCCESS =
    (SELECT LASTSUCCESS FROM ASN.IBMSNAP_SUBS_SET B
     WHERE APPLY_QUAL = 'ApplyQual' AND
           SET_NAME = 'ExistName' AND
```



```

                                WHOS_ON_FIRST = 'Val')
WHERE
    APPLY_QUAL      = 'ApplyQual' AND
    SET_NAME        = 'NewName'    AND
    WHOS_ON_FIRST   = 'Val';

```

where *ApplyQual* is the Apply qualifier, *ExistName* is the name of the existing subscription set that is being split, *Val* is either F or S, and *NewName* is the name of the new subscription set that you are creating.

4. From the Capture control server, run the following SQL statement to insert a new row for the new subscription set into the prune set (IBMSNAP\_PRUNE\_SET) table:

```

INSERT INTO Schema.IBMSNAP_PRUNE_SET
    (APPLY_QUALIFIER,
     SET_NAME,
     TARGET_SERVER,
     SYNCHTIME,
     SYNCHPOINT
VALUES ('ApplyQual',
       'NewName',
       'Target_Server',
       NULL,
       x'000000000000000000000000');

```

where *Schema* is the name of the Capture schema, *ApplyQual* is the Apply qualifier, *NewName* is the name of the new subscription set that you are creating, and *Target\_Server* is the database location of the target tables.

5. From the Capture control server, run the following SQL statement to copy information from the existing subscription set row to the new subscription set row in the IBMSNAP\_PRUNE\_SET table:

```

UPDATE Schema.IBMSNAP_PRUNE_SET
    SET SYNCHPOINT =
        (SELECT SYNCHPOINT FROM Schema.IBMSNAP_PRUNE_SET B
         WHERE APPLY_QUAL      = 'ApplyQual' AND
               SET_NAME        = 'ExistName' AND
               TARGET_SERVER   = 'Target_Server'),
    SYNCHTIME =
        (SELECT SYNCHTIME FROM Schema.IBMSNAP_PRUNE_SET B
         WHERE APPLY_QUAL      = 'ApplyQual' AND
               SET_NAME        = 'ExistName' AND
               TARGET_SERVER   = 'Target_Server')
WHERE
    APPLY_QUAL      = 'ApplyQual' AND
    SET_NAME        = 'NewName'    AND
    TARGET_SERVER   = 'Target_Server';

```

where *Schema* is the name of the Capture schema, *ApplyQual* is the Apply qualifier, *ExistName* is the name of the existing subscription set that is being split, *Target\_Server* is the database location of the target tables, and *NewName* is the name of the new subscription set that you are creating.

6. From the Apply control server, run the following SQL statements to change the subscription set name in the subscription members (IBMSNAP\_SUBS\_MEMBR) and the subscription columns (IBMSNAP\_SUBS\_COLS) tables for *each* subscription-set member that you are moving into the new subscription set:

```
UPDATE ASN.IBMSNAP_SUBS_MEMBR
  SET SET_NAME      = 'NewName'
 WHERE
      APPLY_QUAL      = 'ApplyQual' AND
      SET_NAME        = 'ExistName' AND
      WHOS_ON_FIRST   = 'Val'       AND
      SOURCE_OWNER     = 'SrcSchema' AND
      SOURCE_TABLE     = 'SrcTbl'    AND
      SOURCE_VIEW_QUAL = 'SrcVwQual' AND
      TARGET_OWNER     = 'TgtSchema' AND
      TARGET_TABLE     = 'TgtTbl';

UPDATE ASN.IBMSNAP_SUBS_COLS
  SET SET_NAME      = 'NewName'
 WHERE
      APPLY_QUAL      = 'ApplyQual' AND
      SET_NAME        = 'ExistName' AND
      WHOS_ON_FIRST   = 'Val'       AND
      TARGET_OWNER     = 'TgtSchema' AND
      TARGET_TABLE     = 'TgtTbl';
```

where *NewName* is the name of the new subscription set that you are creating, *ApplyQual* is the Apply qualifier, *ExistName* is the name of the existing subscription set being split, *Val* is either F or S, *SrcSchema* is the source table schema, *SrcTbl* is the source table name, *SrcVwQual* is the source-view qualifier for this source table, *TgtSchema* is the schema of the target table, and *TgtTbl* is the target table name.

Repeat this step for each subscription-set member that you want to move to the new subscription set.

7. If the subscription set that you are splitting uses before or after SQL statements or procedure calls, move the applicable statements to the new subscription set in the subscription statements (IBMSNAP\_SUBS\_STMTS) table:

- a. Run the following SQL script from the Apply control server to move the statements:

```
UPDATE ASN.IBMSNAP_SUBS_STMTS
  SET SET_NAME      = 'NewName'
 WHERE
      APPLY_QUAL      = 'ApplyQual' AND
      SET_NAME        = 'ExistName' AND
      WHOS_ON_FIRST   = 'Val'       AND
      SMT_NUMBER      in (Stmt1, Stmt2, .. Stmtn);
```

where *NewName* is the name of the new subscription set that you are creating, *ApplyQual* is the Apply qualifier, *ExistName* is the name of the existing subscription set being split, *Val* is either F or S, and *Stmt1*, *Stmt2*, and *Stmtn* correspond to the numbers of the statements that you are moving to the new subscription set.

- b. Adjust the AUX\_STMTS column values in the IBMSNAP\_SUBS\_SET table to reflect the new count of statements for both subscription sets. Renumber the statements to eliminate any duplicates, if necessary.
8. From the Capture control server, run the following SQL statement to change the name of the subscription set in the pruning control (IBMSNAP\_PRUNCNTL) table for *each* subscription-set member that you moved:

```
UPDATE Schema.IBMSNAP_PRUNCNTL
  SET SET_NAME      = 'NewName'
 WHERE
    APPLY_QUAL      = 'ApplyQual'      AND
    SET_NAME        = 'ExistName'      AND
    TARGET_SERVER   = 'Target_Server'  AND
    SOURCE_OWNER    = 'SrcSchema'      AND
    SOURCE_TABLE    = 'SrcTbl'         AND
    SOURCE_VIEW_QUAL = SrcVwQual       AND
    TARGET_OWNER    = 'TgtSchema'      AND
    TARGET_TABLE    = 'TgtTbl';
```

where *Schema* is the name of the Capture schema, *NewName* is the name of the new subscription set that you created in step 2, *ApplyQual* is the Apply qualifier, *ExistName* is the name of the existing subscription set that was split, *Target\_Server* is the database location of the target tables, *SrcSchema* is the source table schema, *SrcTbl* is the source table name, *SrcVwQual* is the source-view qualifier for this replication source table, *TgtSchema* is the target table schema, and *TgtTbl* is the target table name.

Repeat this step for each subscription-set member that you moved to the new subscription set.

9. **For UNIX, Windows, z/OS:** If you are running the Apply program with **opt4one** set to y, stop and then restart the Apply program.
10. Reactivate both subscription sets using the following method:

### Replication Center

From the Subscription Sets folder, right-click both deactivated subscription sets in the contents pane and select Activate. See the Replication Center help for details.

---

## Merging subscription sets

If you want to merge two subscription sets into one, you can use the following procedure. You might want to merge subscription sets if you want the target tables within these two subscription sets to have the same transaction consistency but you do not want to delete and then re-create subscription set information.

### Prerequisites:

Before running these SQL statements, familiarize yourself with the structure of the DB2 replication control tables and with the subscription sets defined on your system.

Identify the Capture control server, target server, and Apply control server of each subscription set that you want to merge. Verify that all of the subscription sets that you want to merge were created with the same Capture control server, target server, and Apply control server.

**For UNIX, Windows, z/OS:** If you set up monitoring definitions or started Replication Alert Monitor programs to detect alert conditions for the subscription sets that you are going to merge, drop these monitoring definitions. After you merge the subscription sets, re-create the monitoring definitions through the Replication Center. Then, you can reinitialize the associated Replication Alert Monitor programs using the **asnmcmd** system command with the **reinit** parameter. Alternatively, you can stop the Replication Alert Monitor programs using the **asnmcmd** system command with the **stop** parameter and then restart the programs using the **asnmon** system command.

### Restriction:

The two subscription sets that you want to merge *must* derive their source data from the same Capture server and through the same Capture schema.

### Procedure:

1. Use one of the following methods to stop the associated Capture program:

#### Replication Center

Use the Stop Capture window. See the Replication Center help for details.

#### asnccmd system command (Windows, UNIX, z/OS)

Use the **stop** parameter. See “asnccmd: Operating Capture (UNIX, Windows, z/OS)” on page 322 for parameter descriptions and command syntax.

## ENDDPRCAP system command (OS/400)

See “ENDDPRCAP: Stopping Capture (OS/400)” on page 400 for parameter descriptions and command syntax.

Wait until both subscription sets reach the same synchpoint and synctime as indicated in the subscription sets (IBMSNAP\_SUBS\_SET) table.

**Important:** The two subscription sets *must* have processed the source data up to the identical synchpoint value to prevent a loss of data when the subscription sets are merged.

**Tip:** If you do not want to stop the Capture program, insert a USER signal in the signal (IBMSNAP\_SIGNAL) table, and generate an event with the END\_SYNCHPOINT (in the subscriptions events [IBMSNAP\_SUBS\_EVENT] table) set to the value of the SIGNAL\_LSN column in the IBMSNAP\_SIGNAL table so that only the data up to that end point is applied.

2. Use the following method to deactivate the two subscription sets:

### Replication Center

From the Subscription Sets folder, right-click the two active subscription sets in the contents pane and select Deactivate. See the Replication Center help for details.

3. From the Apply control server, run the following SQL statement to delete the row from the IBMSNAP\_SUBS\_SET table that corresponds to the subscription set that you are moving into the other subscription set:

```
DELETE FROM ASN.IBMSNAP_SUBS_SET
WHERE
    APPLY_QUAL      = 'ApplyQual'      AND
    SET_NAME        = 'Subset_To_Move' AND
    WHOS_ON_FIRST   = 'Val';
```

where *ApplyQual* is the Apply qualifier, *Subset\_To\_Move* is the name of the subscription set that you are moving into another existing subscription set, and *Val* is either F or S.

4. From the Capture control server, run the following SQL statement to delete the row from the prune set (IBMSNAP\_PRUNE\_SET) table that corresponds to the subscription set that you are moving into the other subscription set:

```
DELETE FROM Schema.IBMSNAP_PRUNE_SET
WHERE
    APPLY_QUAL      = 'ApplyQual'      AND
    SET_NAME        = 'Subset_To_Move' AND
    TARGET_SERVER   = 'Target_Server' ;
```

where *Schema* is the name of the Capture schema, *ApplyQual* is the Apply qualifier, *Subset\_To\_Move* is the name of the subscription set that you are moving into another existing subscription set, and *Target\_Server* is the database location of the target tables.

5. From the Apply control server, run the following SQL statements to change the name of the subscription set that you are moving to the name of the other subscription set in the subscription members (IBMSNAP\_SUBS\_MEMBR) and subscription columns (IBMSNAP\_SUBS\_COLS) tables:

```
UPDATE ASN.IBMSNAP_SUBS_MEMBR
  SET SET_NAME = 'Existing_Merged_Subset'
 WHERE
    APPLY_QUAL = 'ApplyQual' AND
    SET_NAME = 'Subset_To_Move' AND
    WHOS_ON_FIRST = 'Val';

UPDATE ASN.IBMSNAP_SUBS_COLS
  SET SET_NAME = 'Existing_Merged_Subset'
 WHERE
    APPLY_QUAL = 'ApplyQual' AND
    SET_NAME = 'Subset_To_Move' AND
    WHOS_ON_FIRST = 'Val';
```

where *Existing\_Merged\_Subset* is the name of the existing subscription set being merged with the subscription set that you are moving, *ApplyQual* is the Apply qualifier, *Subset\_To\_Move* is the name of the subscription set that you are moving into the existing subscription set, and *Val* is either F or S.

6. If the subscription set that you are moving uses before or after SQL statements or procedure calls, change the name of the subscription set in the subscription statements (IBMSNAP\_SUBS\_STMTS) table:
  - a. Run the following SQL script from the Apply control server to change the name of the subscription set:

```
UPDATE ASN.IBMSNAP_SUBS_STMTS
  SET SET_NAME = 'Existing_Merged_Subset'
 WHERE
    APPLY_QUAL = 'ApplyQual' AND
    SET_NAME = 'Subset_To_Move' AND
    WHOS_ON_FIRST = 'Val';
```

where *Existing\_Merged\_Subset* is the name of the existing subscription set that is being merged with the subscription set that you are moving, *ApplyQual* is the Apply qualifier, *Subset\_To\_Move* is the name of the subscription set that you are moving into the existing subscription set, and *Val* is either F or S.

- b. Adjust the AUX\_STMTS column value in the IBMSNAP\_SUBS\_SET table to reflect the new count of statements in the existing merged subscription set. Renumber the statements to eliminate any duplicates, if necessary.

7. From the Capture control server, run the following SQL statement to change the name of the subscription set that was moved to the name of the merged subscription set in the pruning control (IBMSNAP\_PRUNCNTL) table:

```
UPDATE Schema.IBMSNAP_PRUNCNTL
  SET SET_NAME      = 'Existing_Merged_Subset'
WHERE
  APPLY_QUAL      = 'ApplyQual'      AND
  SET_NAME        = 'Subset_To_Move' AND
  TARGET_SERVER   = 'Target_Server' ;
```

where *Schema* is the name of the Capture schema, *Existing\_Merged\_Subset* is the name of the existing subscription set being merged with the subscription set that you are moving, *ApplyQual* is the Apply qualifier, *Subset\_To\_Move* is the name of the subscription set that you are moving into another existing subscription set, and *Target\_Server* is the database location of the target tables.

8. **For UNIX, Windows, z/OS:** If you are running the Apply program with **opt4one** set to y, stop and then restart the Apply program.
9. Reactivate the merged subscription set using the following method:

#### **Replication Center**

From the Subscription Sets folder, right-click the deactivated subscription set in the contents pane and select Activate. See the Replication Center help for details.

---

## **Changing Apply qualifiers of subscription sets**

If you need to change the Apply qualifier of a subscription set, you can use SQL to make the change without deleting and re-creating the subscription set.

If you have several subscription sets using the same Apply qualifier, you might want to move some of the subscription sets to a new Apply qualifier to balance the work loads of the Apply programs.

You must run the SQL statements in this procedure for *each* subscription set that you want to move.

#### **Prerequisites:**

Before running these SQL statements, familiarize yourself with the structure of the DB2 replication control tables and with the subscription sets defined on your system.

You must also determine the following information:

- The name of the new Apply qualifier. (See Chapter 16, “Naming rules for replication objects” on page 301 for more information.)
- The subscription sets that you want to move from the existing Apply qualifier to the new Apply qualifier.
- Any before or after SQL statements or procedure calls that are defined for these subscription sets.

**For UNIX, Windows, z/OS:** If you set up monitoring definitions or started Replication Alert Monitor programs under the Apply qualifier that you are going to change, drop these monitoring definitions. After you change the Apply qualifier, re-create the monitoring definitions with the new Apply qualifier name through the Replication Center. Then, you can reinitialize the associated Replication Alert Monitor programs using the **asnmcmd** system command with the **reinit** parameter. Alternatively, you can stop the Replication Alert Monitor programs using the **asnmcmd** system command with the **stop** parameter and then restart the programs using the **asnmmon** system command.

#### Procedure:

1. Deactivate the subscription sets that you want to change using the following method:

##### Replication Center

From the Subscription Sets folder, right-click the active subscription sets in the contents pane and select Deactivate. See the Replication Center help for details.

2. From the Apply control server, run the following SQL statements to change the Apply qualifier of the subscription set in the subscription sets (IBMSNAP\_SUBS\_SET), subscription members (IBMSNAP\_SUBS\_MEMBR), and subscription columns (IBMSNAP\_SUBS\_COLS) tables:

```
UPDATE ASN.IBMSNAP_SUBS_SET
  SET APPLY_QUAL = 'NewApplyQual'
 WHERE
      APPLY_QUAL = 'ExistApplyQual' AND
      SET_NAME   = 'Name'             AND
      WHOS_ON_FIRST = 'Val';

UPDATE ASN.IBMSNAP_SUBS_MEMBR
  SET APPLY_QUAL = 'NewApplyQual'
 WHERE
      APPLY_QUAL = 'ExistApplyQual' AND
      SET_NAME   = 'Name'             AND
      WHOS_ON_FIRST = 'Val';
```



```

UPDATE ASN.IBMSNAP_SUBS_COLS
  SET APPLY_QUAL = 'NewApplyQual'
WHERE
  APPLY_QUAL = 'ExistApplyQual' AND
  SET_NAME = 'Name' AND
  WHOS_ON_FIRST = 'Val';

```

where *NewApplyQual* is the new Apply qualifier, *ExistApplyQual* is the existing Apply qualifier, *Name* is the name of the subscription set, and *Val* is either F or S.

3. If this subscription set uses before or after SQL statements or procedure calls, run the following SQL script from the Apply control server to change the Apply qualifier of the subscription set in the subscription statements (IBMSNAP\_SUBS\_STMTS) table:

```

UPDATE ASN.IBMSNAP_SUBS_STMTS
  SET APPLY_QUAL = 'NewApplyQual'
WHERE
  APPLY_QUAL = 'ExistApplyQual' AND
  SET_NAME = 'Name' AND
  WHOS_ON_FIRST = 'Val';

```

where *NewApplyQual* is the new Apply qualifier, *ExistApplyQual* is the existing Apply qualifier, *Name* is the name of the subscription set, and *Val* is either F or S.

4. From the Capture control server, run the following SQL statements to change the Apply qualifier of the subscription set in the prune set (IBMSNAP\_PRUNE\_SET) and pruning control (IBMSNAP\_PRUNCNTL) tables:

```

UPDATE Schema.IBMSNAP_PRUNE_SET
  SET APPLY_QUAL = 'NewApplyQual'
WHERE
  APPLY_QUAL = 'ExistApplyQual' AND
  SET_NAME = 'Name' AND
  TARGET_SERVER = 'Target_Server';

UPDATE Schema.IBMSNAP_PRUNCNTL
  SET APPLY_QUAL = 'NewApplyQual'
WHERE
  APPLY_QUAL = 'ExistApplyQual' AND
  SET_NAME = 'Name' AND
  TARGET_SERVER = 'Target_Server';

```

where *Schema* is the name of the Capture schema, *NewApplyQual* is the new Apply qualifier, *ExistApplyQual* is the existing Apply qualifier, *Name* is the name of the subscription set, and *Target\_Server* is the database location of the target tables.

5. Repeat steps 2 through 4 for each remaining subscription set that you want to move.

6. **For UNIX, Windows, z/OS:** If you are running the Apply program with **opt4one** set to **y**, stop and then restart the Apply program.
7. Use the following method to reactivate the subscription sets:

#### **Replication Center**

From the Subscription Sets folder, right-click the deactivated subscription sets in the contents pane and select **Activate**. See the Replication Center help for details.

---

## **Deactivating subscription sets**

You can deactivate a subscription set without removing it. When you deactivate a subscription set, the Apply program completes its current processing cycle and then suspends operations for that subscription set. You might need to perform special maintenance on these deactivated subscription sets depending on how long they must remain deactivated:

### **Short time-period**

There are no special processing requirements for subscription sets that you temporarily deactivate. You should temporarily deactivate a subscription set while changing its attributes or while fixing failures on target tables.

Use the Replication Center to deactivate, change, and then reactivate a subscription set.

### **Longer time-period**

You can deactivate a subscription set that you do not currently need but might want to use in the future. However, you must take additional action if this subscription set needs to remain deactivated for a time period that is long enough for changed data to accumulate and to affect the performance of the Capture and Apply programs.

The Capture program uses information from active Apply programs during the pruning process. If the Apply programs are inactive or the subscriptions sets are deactivated for long periods of time, the pruning information becomes stale and the unit-of-work (UOW) and possibly the change-data (CD) tables cannot be pruned quickly and efficiently if active registrations that are associated with the deactivated subscription sets remain. This stale information can seriously degrade the performance of the remaining active Apply programs and cause unnecessary and costly CPU consumption by the pruning process. The UOW and CD tables are eventually pruned based on the retention limit (with a default value of seven days) of the Capture program. However, large amounts of data might accumulate during this time depending on the size of your replication environment.

To prevent these pruning problems, you can use SQL to reset the pruning information for a subscription set that must remain deactivated for a longer time-period.

**Prerequisite:**

Before running these SQL statements, familiarize yourself with the structure of the DB2 replication control tables and with the subscription sets defined on your system.

**Procedure:**

1. From the Replication Center, verify that the subscription set is not active.
2. From the Capture control server, run the following SQL statements to reset the pruning information in the prune set (IBMSNAP\_PRUNE\_SET) and pruning control (IBMSNAP\_PRUNCNTL) tables for the deactivated subscription set:

```
UPDATE Schema.IBMSNAP_PRUNE_SET
  SET SYNCHPOINT = x'00000000000000000000' AND
      SYNCHTIME   = NULL
WHERE
  APPLY_QUAL = 'ApplyQual' AND
  SET_NAME   = 'Name'       AND
  TARGET_SERVER = 'Target_Server';

UPDATE Schema.IBMSNAP_PRUNCNTL
  SET SYNCHPOINT = NULL AND
      SYNCHTIME   = NULL
WHERE
  APPLY_QUAL = 'ApplyQual' AND
  SET_NAME   = 'Name'       AND
  TARGET_SERVER = 'Target_Server';
```

where *Schema* is the name of the Capture schema, *ApplyQual* is the Apply qualifier, *Name* is the name of the subscription set, and *Target\_Server* is the database location of the target tables.

If you deactivated all the subscription sets associated with a registered object, you should also deactivate the registered object to prevent the Capture program from capturing data unnecessarily.

---

## Removing subscription sets

If you no longer need to replicate the data in a particular subscription set, you can remove the subscription set. However, if your Apply program is processing the subscription set that you remove, your Apply program job abends and any other subscription sets in that job are not processed until you restart the job.

**Procedure:**

1. To ensure that the Apply program has completed any current processing for the subscription set, deactivate the subscription set before you remove it by using the following method:

**Replication Center**

From the Subscription Sets folder, right-click the active subscription set in the contents pane and select Deactivate. See the Replication Center help for details.

2. Use one of the following methods to remove a deactivated subscription set:

**Replication Center**

Use the Delete Subscription Set window. See the Replication Center help for details.

**RMVDPRSUB system command (OS/400)**

See “RMVDPRSUB: Removing a DPR subscription set (OS/400)” on page 421 for parameter descriptions and command syntax.

**Important:** The Capture program continues capturing data and writing rows to the change-data (CD) table even if you remove all subscription sets for the registered object. To prevent this continued processing by the Capture program, deactivate or remove the registered object after removing its subscription sets.

---

**Coordinating replication events with database application events**

You can coordinate database and replication events by manually inserting rows into the signal (IBMSNAP\_SIGNAL) table. Manually inserted IBMSNAP\_SIGNAL rows, known as signals, instruct running Capture programs to take specific actions.

**Setting an event END\_SYNCHPOINT using the USER type signal**

You can set the SIGNAL\_TYPE column value to USER to establish a precise point on the DB2 recovery log and to coordinate a replication event with a database application event.

For example, if you are replicating online transaction processing (OLTP) data to a separately maintained data warehouse, you might want to keep the warehouse data fairly stable for ad hoc query processing. So you update the warehouse data with only the changes that occurred up to a specific point in time in the OLTP application business day. In this case, the database application event is the logical end of the business day. The replication event would be the application of the changes from the close of business on one specific day to the close of business on the following day. Assume that the subscription sets are configured for event processing only.

## Procedure:

To create a USER type signal:

1. Create a Capture USER type signal by inserting the following row into the IBMSNAP\_SIGNAL table:

```
INSERT INTO Schema.IBMSNAP_SIGNAL
    (signal_type,
     signal_subtype,
     signal_state)
VALUES('USER',
      'USER APPLY EVENT SIGNAL',
      'P');
```

Run this SQL INSERT statement when the database application event occurs (in this case at the end of the application business day).

The Capture program acts on this signal table log record after the Capture program finds this record on the database recovery log and only when the Capture program finds the corresponding commit record for this insert, verifying that this event was committed.

When a USER type signal is committed, the Capture program updates the following IBMSNAP\_SIGNAL column values that correspond to the insert log record being processed:

- SIGNAL\_STATE = 'R' (received by the Capture program)
  - SIGNAL\_LSN = the log sequence number from the commit log record for the DB2 unit of work that contains this signal row insert
2. Use the value that is now in the SIGNAL\_LSN column of the inserted signal row as an END\_SYNCHPOINT value in the subscription events (IBMSNAP\_SUBS\_EVENT) control table. This new value alerts the Apply program that all the data for the new business day has been collected by the Capture program and that the Apply program should fetch and apply data only up to the value of the SIGNAL\_LSN column.

You can automate the insert into the IBMSNAP\_SUBS\_EVENT table by creating an update trigger on the IBMSNAP\_SIGNAL table:

```
CREATE TRIGGER EVENT_TRIG
NO CASCADE AFTER UPDATE ON Schema.IBMSNAP_SIGNAL
REFERENCING NEW AS N
FOR EACH ROW MODE DB2SQL
WHEN (N.SIGNAL_SUBTYPE = 'USER APPLY EVENT SIGNAL')
INSERT INTO ASN.IBMSNAP_SUBS_EVENT VALUES
    ('WH_APPLY_EVENT',
     (CURRENT_TIMESTAMP + 2 MINUTES),
     N.SIGNAL_LSN,
     null);
```

This trigger fires each time that the IBMSNAP\_SIGNAL table is updated by the Capture program. When a SIGNAL\_SUBTYPE column is updated to 'USER APPLY EVENT SIGNAL', the trigger inserts a row into the IBMSNAP\_SUBS\_EVENT table. This row indicates to the Apply program that it must fetch and apply the work from the latest business day (which has been committed prior to the SIGNAL\_LSN value as computed by the Capture program) after two minutes have elapsed.

## Using the Capture CMD STOP signal

You can set the SIGNAL\_TYPE column value to CMD and the SIGNAL\_SUBTYPE column value to STOP to stop a Capture program process at a precise point on the DB2 recovery log. There are two main uses of this capability:

- To coordinate the Capture program with any source table changes that render previous log records unreadable. This could occur if you dropped and then re-created a table or if you reorganized a table without setting the KEEPDICTIONARY option to YES.
- To coordinate a common recovery point between replicated distributed database systems.

### Coordinating a source table change with the Capture program

You can use a Capture CMD type STOP subtype signal to shut down a Capture program and to coordinate source table changes.

#### Procedure:

To coordinate the source table changes:

1. Create a Capture CMD type STOP subtype signal by inserting a row into the signal (IBMSNAP\_SIGNAL) table using the following SQL statement:

```
INSERT INTO Schema.IBMSNAP_SIGNAL
    (signal_type,
     signal_subtype,
     signal_state)
VALUES('CMD',
      'STOP',
      'P');
```

You should insert this row when the database application event occurs, after the source table activity has been quiesced but prior to the activity that causes problematic log record changes.

The Capture program acts on this signal table log record after the Capture program finds this record on the database recovery log and only when the Capture program finds the corresponding commit record for this insert, verifying that this event was committed.

The Capture program shuts down all Capture threads in an orderly manner after committing all captured data from the transactions on the log that are prior to the commit log record for the DB2 unit of work that contains this inserted IBMSNAP\_SIGNAL row. Before terminating, the Capture program also updates the following values in the IBMSNAP\_SIGNAL table row that corresponds to the insert log record being processed:

- SIGNAL\_STATE = 'R' (received by the Capture program)
- SIGNAL\_LSN = the log sequence number from the commit log record for the DB2 unit of work that contains this signal row insert

All log records for the changing source table are processed by the Capture program when it terminates.

2. Depending on your scenario, drop and re-create your source table, or reorganize and compress your source table without set the KEEPDICTIONARY option to YES.
3. If you dropped or altered replicated columns, you should now alter the corresponding registrations and subscription sets that you created for this source table. Such changes, if necessary, can be further coordinated with the Apply program by waiting for the affected subscription sets to catch up to the currently stopped Capture program. A subscription set is in synch with the Capture program when the SYNCHPOINT column value in the subscription sets (IBMSNAP\_SUBS\_SET) table is equal to the MAX\_COMMITSEQ column value in the *Schema*.IBMSNAP\_RESTART table.

### Setting a distributed recovery point

You can use a Capture CMD type STOP subtype signal to set your source and target databases to equivalent recovery points and recover the databases at a common point of consistency.

### Prerequisites:

Before using this procedure, verify that your Apply control tables have been created in the target database.

Also, verify that all activity against the source database has been quiesced before inserting the row into the IBMSNAP\_SIGNAL table. However, do not create the backup or image copy of the database tables until after you insert the row into the IBMSNAP\_SIGNAL table.

If your subscription sets are not typically configured for event processing, then you must temporarily set your subscription sets to event-based timing. Use the following SQL statement to insert a row into the subscription events (IBMSNAP\_SUBS\_EVENT) table:

```

INSERT INTO ASN.IBMSNAP_SUBS_EVENT
VALUES('RECOVERY_EVENT',
      CURRENT_TIMESTAMP + 2 MINUTES,
      SIGNAL_LSN_value,
      NULL);

```

where *SIGNAL\_LSN\_value* is the log sequence number set by the Capture program and stored in the IBMSNAP\_SIGNAL table.

### Procedure:

To set a distributed recovery point:

1. Create a Capture CMD type STOP subtype signal by inserting a row into the IBMSNAP\_SIGNAL table using the following SQL statement:

```

INSERT INTO Schema.IBMSNAP_SIGNAL
(signal_type,
 signal_subtype,
 signal_state)
VALUES('CMD',
      'STOP',
      'P');

```

The Capture program acts on this signal table log record after the Capture program finds this record on the database recovery log and only when the Capture program finds the corresponding commit record for this insert, verifying that this event was committed.

The Capture program shuts down all Capture threads in an orderly manner after committing all captured data from the transactions on the log that is prior to the commit log record for the DB2 unit of work that contains this inserted IBMSNAP\_SIGNAL row. Before terminating, the Capture program also updates the following values in the IBMSNAP\_SIGNAL table row that corresponds to the insert log record being processed:

- SIGNAL\_STATE = 'R' (received by the Capture program)
- SIGNAL\_LSN = the log sequence number from the commit log record for the DB2 unit of work that contains this signal row insert

All log records for the source database are processed by the Capture program when it terminates.

2. Run the source database backup or image copy utilities.
3. Use the value in the SIGNAL\_LSN column from the IBMSNAP\_SIGNAL table row that you inserted as an END\_SYNCHPOINT value in the IBMSNAP\_SUBS\_EVENT table. This value alerts the Apply program that all the data committed prior to the backup point has been collected by the



Capture program and that the Apply program should fetch and apply data only up to the value of the SIGNAL\_LSN column.

The subscription sets process all data up to the SIGNAL\_LSN value.

4. Run the target database backup or image copy utilities. The source and target databases now have equivalent recovery points, and you can recover both databases at a common point of consistency.

You can resume all source database activity as soon as the Apply events have been set and the source database backup or image copy utility activity is complete. You can also start the Capture program. After the target database backup or image copy utility activity is complete, you can change the scheduling options of your subscription sets back to their original settings (time-based, event-based, or both).

### Performing a CAPSTART handshake signal outside of the Apply program

Before any subscription set can be used by the Apply program to fetch and apply changes from the CD tables, there must be a "handshake" (synchronized communication) between the Capture and Apply programs of each subscription-set member in that subscription set.

The Apply program initiates the handshake by inserting a CMD type CAPSTART subtype signal into the signal (IBMSNAP\_SIGNAL) table. The Apply program inserts this signal before performing a full refresh of any subscription-set member with a target table that is defined as complete.

#### Procedure:

To perform a CAPSTART handshake signal:

- Create a Capture CMD type CAPSTART subtype signal by inserting a row into the IBMSNAP\_SIGNAL table using the following SQL statement:

```
INSERT INTO Schema.IBMSNAP_SIGNAL
    (signal_type,
     signal_subtype,
     signal_input_in,
     signal_state)
VALUES('CMD',
      'CAPSTART',
      mapid,
      'P');
```

where *mapid* is the MAP\_ID column value of the *Schema.IBMSNAP\_PRUNCNTL* table and corresponds to the row for the subscription-set member requiring the handshake.

**Note:** Run this SQL INSERT statement before performing a full refresh of the subscription-set member, if necessary.

The Capture program acts on this signal table log record after the Capture program finds this record on the database recovery log and only when the Capture program finds the corresponding commit record for this insert, verifying that this event was committed.

The Capture program checks if it already placed the associated registration into memory based on prior use of the registered table. If the registered table is not in use, the Capture program reads the associated registration information into memory and sets values in the register (IBMSNAP\_REGISTER) table to show that this registered table is now active and in use.

Regardless of whether or not the registered table is in use, the Capture program sets the values of the SYNCHPOINT and SYNCHTIME columns in the associated row in the *Schema*.IBMSNAP\_PRUNCNTL table to the log sequence number from the commit log record for the DB2 unit of work that contains this inserted signal row and to the timestamp from that same commit log record, respectively.

The Capture program updates the following values in the IBMSNAP\_SIGNAL table row that corresponds to the insert log record being processed:

- SIGNAL\_STATE = 'C' (received and completed by the Capture program)
- SIGNAL\_LSN = the log sequence number from the commit log record for the DB2 unit of work that contains this signal row insert

## Performing a CAPSTOP signal

You can initiate a CAPSTOP signal if you want to manually stop capturing changes for a registration. You can use this signal when deactivating a registration or before you remove a registration.

### Procedure:

To perform a CAPSTOP signal:

1. Create a Capture CMD type CAPSTOP subtype signal by inserting a row into the IBMSNAP\_SIGNAL table using the following SQL statement:

```
INSERT INTO Schema.IBMSNAP_SIGNAL
    (signal_type,
     signal_subtype,
     signal_input_in,
     signal_state)
VALUES('CMD',
      'CAPSTOP',
      source_owner.source_table,
      'P');
```

where *Schema* is the name of the Capture schema and *source\_owner.source\_table* is the fully qualified name of the table that no longer requires captured changes.

The Capture program acts on this signal table log record after the Capture program finds this record on the database recovery log and only when the Capture program finds the corresponding commit record for this insert, verifying that this event was committed.

The Capture program checks if it has already placed the associated registration into memory based on prior use of the registered table. If the registered table is not currently in use, the Capture program ignores the CAPSTOP signal.

If the registered table is in use, the Capture program clears the memory associated with this registration and inactivates the registration (by setting the STATE column in the IBMSNAP\_REGISTER table to 'I'). The Capture program then stops capturing changes for this registered table.

The Capture program updates the following column values in the IBMSNAP\_SIGNAL table row that corresponds to the insert log record being processed:

- SIGNAL\_STATE = 'C' (received and completed by the Capture program)
  - SIGNAL\_LSN = the log sequence number from the commit log record for the DB2 unit of work that contains this signal row insert
2. Optional: Remove the registration.

---

## Promoting your replication configuration to another system

When you define registered objects or subscription sets on one system (a test system, for example), and you need to copy the replication environment to another system (a production system, for example), you can use the promote functions of the Replication Center. These functions reverse engineer your registered objects or subscription sets to create script files with appropriate data definition language (DDL) and data manipulation language (DML). You can copy the replication definitions to another database without having to re-register the sources or re-create the subscription sets.

For example, use the promote functions to define subscription sets for remote target databases. After you define a model target system in your test environment, you can create subscription-set scripts (and modify which Apply qualifier is used and so on) for your remote target systems, which are not otherwise supported from a central control point.

**Important:** The promote functions do not connect to the destination target system and do not validate the replication configuration parameters of that system.

There are three promote functions:

#### **Promote registered tables**

This function promotes registration information for specified tables. This function optionally promotes base table, index and table space definitions. You can specify a different Capture schema and a different server name for the tables that you promote. Also, you can change the schema name for the change-data (CD) tables associated with the promoted source tables.

You can promote multiple registered tables at one time. The new schema names that you provide are applied to all the promoted tables.

This function promotes tables that are registered under DB2 Universal Database Version 8 only.

#### **Promote registered views**

This function promotes registration information for specified views. This function optionally promotes base view, unregistered base table (on which a view is based), index, and table space definitions. You can specify a different Capture schema and a different server name for the views that you promote. Also, you can change the schema name for the CD views that are associated with the promoted source views and the CD tables on which these CD views are based.

You can promote multiple registered views at one time. The new schema names that you provide are applied to all the promoted views.

**Important:** If the view that you are promoting is based on a registered source table, you must promote the registered source table separately by using the promote registered tables function. These registered source tables are not automatically promoted by the promote registered view function. However, the unregistered base tables, upon which this view is based, are promoted by this function, if required.

#### **Promote subscription sets**

This function promotes subscription sets. This function enables you to copy a subscription set (with all of its subscription-set members) from one database to another.

You should use the promote subscription sets function with the promote registered tables function.

**Important:** You can use the promote functions to promote registered objects and subscription sets that reside on OS/400, UNIX, Windows, and z/OS operating systems. The promote functions copy replication definitions between like systems only, for example from one DB2 Universal Database for z/OS system to another DB2 Universal Database for z/OS system.

You cannot use the promote functions to copy replication definitions to or from non-DB2 relational databases. Additionally, you cannot use the promote functions to copy replication definitions that include OS/400 remote journals.

**Related concepts:**

- Chapter 14, “Using the DB2 Replication Center” on page 243

**Related tasks:**

- Chapter 3, “Registering tables and views as replication sources” on page 37
- Chapter 4, “Subscribing to sources” on page 63

**Related reference:**

- “*schema*.IBMSNAP\_SIGNAL” on page 507



---

## Chapter 13. Maintaining your replication environment

This chapter explains how to maintain the source systems, control tables, and target tables that reside on your database and are used by DB2 replication.

DB2 replication works with your database system and requires limited changes to your existing database activities. However, to ensure that your entire system continues to run smoothly and to avoid potential problems, you should determine the processing requirements of your replication environment and the potential impact of these requirements on your database system. This chapter discusses the maintenance requirements of these three functional components of DB2 replication:

- “Maintaining your source systems”
- “Maintaining your control tables” on page 231
- “Maintaining your target tables” on page 239

---

### Maintaining your source systems

The replication source system contains the change-capture mechanism, the source tables that you want to replicate (including any remote journals used on OS/400 systems), the log data used by the Capture program, and any capture triggers used on non-DB2 relational database sources. This section explains how to maintain your source tables and log files properly and how to ensure that these tables and files are always accessible to DB2 replication.

#### Maintaining source objects

Replication source objects are database tables and views that require the same maintenance as other database tables and views on your system. Continue to run your existing utilities and maintenance routines on these objects.

You need to consider the availability of these source tables to DB2 replication so that the Capture and Apply programs are always able to proceed. DB2 replication does not require direct access to source tables during most replication processing. However, DB2 replication *must* access your source tables or table spaces directly when one of the following two actions occurs:

- The Apply program performs a full refresh.
- The log manager attempts to read compressed log records (z/OS only).

Make sure that read access is available to your source tables to avoid disrupting your replication Apply program processing during a full refresh. Also, on z/OS, make sure that your utilities run in an online mode so that DB2 can obtain a latch against the compressed log record table space if your

source tables are compressed. If your utilities and maintenance routines run in an exclusive mode that requires your database (or the compressed table space on z/OS) to be taken offline, your source objects will be unavailable to replication.

## **Maintaining and retaining source logs and journal receivers**

Your DB2 recovery logs serve two purposes: to provide DB2 recovery capabilities and to provide information to your running Capture programs. You need to retain log data for both DB2 recovery and for DB2 replication, and you must be absolutely certain that the Capture programs and DB2 are completely finished with a set of logs or journal receivers before you delete this data.

**Note:** DB2 replication does not use log data from non-DB2 relational databases.

### **Retaining log data (UNIX, Windows, z/OS)**

Log data resides in log buffers, active logs, or archive logs. Each time the Capture program warm starts it requires all the DB2 logs created since it stopped as well as any DB2 logs that it did not completely process.

**For UNIX and Windows:** You must configure your database to use user-exit archiving for your Capture programs to retrieve data from archived logs.

If you run the Capture program whenever DB2 is running, the Capture program is typically up to date with the recovery logs of DB2. If you run Capture programs whenever DB2 is up or you retain log records for a week or longer, you can continue to use your existing log retention procedures. However, you should change your log retention procedures to accommodate DB2 replication if:

- You typically delete log records as soon as DB2 completes a backup, and these log records are no longer needed for forward recovery.
- You face storage constraints and need to delete your archived recovery logs frequently.

### **Procedure:**

To determine which log records must be retained for use by the Capture program and which log records can be deleted:

#### **On UNIX and Windows:**

1. Run the following SQL statement to obtain the MIN\_INFLIGHTSEQ value from the restart (IBMSNAP\_RESTART) table:

```
SELECT MIN_INFLIGHTSEQ
FROM ASN.IBMSNAP_RESTART
WITH UR;
```



The MIN\_INFLIGHTSEQ value appears. (There is only one row in the IBMSNAP\_RESTART table.) The MIN\_INFLIGHTSEQ value is a char(10) for bit data column which looks like 20 hexadecimal characters. For example:

```
00000000123456123456
```

Note the *last* 12 characters of the MIN\_INFLIGHTSEQ value. In the example:

```
123456123456
```

2. From a command line, type the **db2 get db cfg** command to obtain the path for the active log files. For example:

```
db2 get db cfg for yourdbname
```

where *yourdbname* is the database name. From the output displayed on the screen, note the path for the active log files. For example:

```
Path to log files    =C:\DB2\NODE0000\SQL00001\SQLLOGDIR\
```

3. From a DB2 command line, type the **db2flsn** command and enter the *last* 12 characters of the MIN\_INFLIGHTSEQ value. For example:

```
C:\DB2\NODE0000\SQL00001\>db2flsn 123456123456
```

To run the **db2flsn** command, you must have access to the SQLLOGCTL.LFH file, which is located one directory above (C:\DB2\NODE0000\SQL00001\ ) the path to active log files.

The system retrieves and displays the name of the file that contains the log record identified by the log sequence number. For example:

```
Given LSN is contained in the log file S000123.LOG
```

4. Note the age of this retrieved log file.  
The Capture program needs this log file and more recent log files to perform a restart at any particular time. You must retain this log file and more recent log files but can delete any older logs to ensure continuous operation of the Capture programs.

#### On z/OS:

1. Run the following SQL statement to obtain the MIN\_INFLIGHTSEQ value from the Restart (IBMSNAP\_RESTART) table:

```
SELECT MIN_INFLIGHTSEQ
FROM ASN.IBMSNAP_RESTART
WITH UR;
```

The MIN\_INFLIGHTSEQ value appears. (There is only one row in the IBMSNAP\_RESTART table.) For example:

000055551F031230000

Ignore the first four characters, which are always 0000. The next 12 characters correspond to the active log sequence number. (This 12-character value is the relative byte address [RBA] in non-data sharing environments and is the log record sequence number [LRSN] in data sharing environments.) The last four characters are 0000 in non-data sharing environments; these last four characters correspond to the member ID in data sharing environments.

2. Use the DSNJU004 utility to invoke the Print Log Map utility. This utility displays information about the bootstrap data sets (BSDS).

For example:

```
# ACTIVE LOG COPY 1 DATA SETS
# START RBA/TIME      END RBA/TIME      DATE      LTIME      DATA SET INFORMATION
#-----
# 555551F03000      555551F05FFF      1998.321  12:48      DSN=DSNC710.LOGCOPY1.DS02
#2001.57 15:46:32.2 2001.057 15:47:03.9 PASSWORD=(NULL)STATUS=TRUNCATED,REUSABLE
# 555551F06000      555551F09FFF      1998.321  12:49      DSN=DSNC710.LOGCOPY1.DS03
#2001.57 15:47:32.2 2001.057 15:48:12.9 PASSWORD=(NULL)STATUS=TRUNCATED,REUSABLE
```

3. Compare your 12-character active log number of the MIN\_INFLIGHTSEQ value to the Start RBA and corresponding End RBA range in each displayed row.
4. Find the row in which the value of your 12-character active log number falls. In the example:

```
# 555551F03000      000001F05FFF      1998.321  12:48      DSN=DSNC710.LOGCOPY1.DS02
#2001.57 15:46:32.2 2001.057 15:47:03.9 PASSWORD=(NULL)STATUS=TRUNCATED,REUSABLE
```

5. Note the corresponding Data Set Information for that active log number. In the example:

DSNC710.LOGCOPY1.DS02

6. Note the date and time of this data set.

The Capture program needs this data set and more recent data sets to perform a restart at any particular time. You must retain this data set and more recent ones but can delete any older data sets to ensure continuous operation of the Capture programs.

Consider the age of this log file or data set to be a benchmark. You must retain this file and more recent log files but can delete any older logs to ensure continuous operation of the Capture programs.

**Recommendation:** Run the Capture program whenever DB2 is up to gain optimal performance, because the Capture program reads the log records directly from the log buffers.

### **Retaining journal receivers (OS/400)**

It is important to retain all journal receivers that are required by the Capture program. When you restart the Capture program with the **RESTART(\*YES)** parameter, the Capture program continues processing from where it ended previously and requires all the journal receivers used by one or more of the source tables.

To make certain your Capture program can access all required journal receivers, use the delete journal receiver exit program, which was registered automatically when you installed DB2 DataPropagator for iSeries. This exit program is invoked any time you or one of your applications programs attempts to delete a journal receiver. This exit program then determines whether or not a journal receiver can be deleted.

**Recommendation:** Specify **DLTRCV(\*YES)** and **MNGRCV(\*SYSTEM)** on the **CHGJRN** or **CRTJRN** command to use the delete journal receiver exit program and leave journal management to the system.

If the journal receiver is used by one or more source tables, the delete journal receiver exit program checks that the receiver being deleted does not contain entries that have not been processed by the Capture program. The exit program *disapproves* the deletion of the receiver if the Capture program still needs to process entries on that receiver. For more information, see “Managing journals and journal receivers (OS/400)” on page 34.

### **Using compression dictionaries (z/OS)**

If you are using DB2 compression dictionary utilities, you must coordinate the use of these utilities with your Capture programs.

### **Updating DB2 compression dictionaries (z/OS)**

When the Capture program requests log records, DB2 must decompress the log records of any table stored in a compressed table space. DB2 uses the current compression dictionary for decompression. If the compression dictionary is temporarily unavailable, DB2 returns an error to the Capture program. The Capture program makes several attempts to continue processing. But if the dictionary remains unavailable, the Capture program issues an ASN0011E message and terminates. Alternatively, if the compression

dictionary is no longer available, the Capture program deactivates the registration. To avoid these situations, let the Capture program process all log records for a table before performing any activity that affects the compression dictionary for that table. These activities include:

- Altering a table space to change its compression setting
- Using DSN1COPY to copy compressed table spaces from one subsystem to another, including from data sharing to non-datasharing environments
- Running the REORG utility on the table space

**Recommendation:** Use the `KEEPDICTIONARY=YES` option to retain the current version of the compression dictionary during a reorganization. The `KEEPDICTIONARY=YES` option ensures that your dictionary remains compatible with your pre-existing log records.

If, however, you prefer to generate a new compression dictionary, synchronize the REORG utility with your current running applications and with the Capture program as follows:

1. Quiesce all application programs that update the table.
2. Let the Capture program capture all logged updates for the table.
3. Use the REORG utility on the compressed table to create a new compression dictionary.
4. Restart your application programs.

### **Latching DB2 compression dictionaries (z/OS)**

You should also consider the availability of your compression directory. When the Capture program reads compressed log records, DB2 takes a latch on the source compressed table space to access the dictionary. The Capture program stops if the compressed table space on the source system is in the STOPPED state when the DB2 Log Read Interface needs this latch. Conversely, a utility that requires complete access to the source table space or that requires the table space to be in a STOPPED state can be locked out by the latch held by the Capture program while it is reading the dictionary.

To prevent any temporary lockout due to an unavailable latch, suspend the Capture program when a source compressed table space needs to be used exclusively by a DB2 (or vendor) utility.

---

## Maintaining your control tables

DB2 replication uses control tables to store source definitions, subscription-set definitions, and other replication-specific control information. Although the size of some control tables remains static, other control tables can grow (and later shrink) dynamically depending on the size of your database and your replication requirements.

The size of the following control tables changes frequently during normal processing:

- Apply job (IBMSNAP\_APPLY\_JOB) (OS/400 only)
- Apply-qualifier cross-reference (IBMSNAP\_AUTHTKN) (OS/400 only)
- Apply trace (IBMSNAP\_APPLYTRACE)
- Apply trail (IBMSNAP\_APPLYTRAIL)
- Capture monitor (IBMSNAP\_CAPMON)
- Capture trace (IBMSNAP\_CAPTRACE)
- Change data (*schema.CD\_table*)
- Consistent-change data (*schema.target\_table*)
- Replication Alert Monitor alerts (IBMSNAP\_ALERTS)
- Replication Alert Monitor trace (IBMSNAP\_MONTRACE)
- Replication Alert Monitor trail (IBMSNAP\_MONTRAIL)
- Signal (IBMSNAP\_SIGNAL)
- Subscription events (IBMSNAP\_SUBS\_EVENT)
- Unit-of-work (IBMSNAP\_UOW)

The size and growth of these dynamic control tables can affect the performance of your system.

This section discusses maintenance activities that you should perform on your control tables.

### Using the RUNSTATS utility (UNIX, Windows, z/OS)

The RUNSTATS utility updates statistics about the physical characteristics of your tables and associated indexes. You should continue to run the RUNSTATS utility on your existing tables at the same frequency as before you used DB2 replication. However, you should run the RUNSTATS utility on your change-data (CD), unit-of-work (IBMSNAP\_UOW), and other dynamic control tables only one time when these tables contain substantial amounts of data. RUNSTATS reports meaningful information about these dynamic tables when these tables are at their maximum production-level size, and the optimizer gains the necessary statistics to determine the best strategy for accessing data.

## Rebinding packages and plans (UNIX, Windows, z/OS)

Many of the DB2 replication packages and plans are bound using isolation UR (uncommitted reads). If you must rebind your packages and plans, note that your internal maintenance programs used for automatic rebinding of these packages and plans can cause contention problems between Capture and Apply if these programs rebind the replication packages with standard options such as cursor stability. DB2 replication packages must remain bound with isolation UR to maintain optimal system performance.

See “Setting up the replication programs” on page 26 for more information.

## Reorganizing your control tables

You should regularly reorganize dynamic control tables that are frequently updated. Your change-data (CD) and unit-of-work (IBMSNAP\_UOW) tables receive many INSERTS during change capture and many DELETES during pruning. The size of the Capture monitor (IBMSNAP\_CAPMON), Capture trace (IBMSNAP\_CAPTRACE), and Apply trail (IBMSNAP\_APPLYTRAIL) tables can change dramatically depending on the update rates of your replication source tables.

### Procedure:

Use one of the following table reorganization methods to eliminate fragmented data and reclaim space:

#### REORG command (UNIX, Windows)

#### REORG utility with the PREFORMAT option (z/OS)

The PREFORMAT option of this utility speeds up the insert processing of the Capture program.

#### RGZPFM (Reorganize Physical File Member) command (OS/400)

You can reorganize the UOW table and active CD tables when the Capture program ends by specifying the **RGZCTLTBL(\*YES)** parameter on the **ENDDPRCAP** command. (See “ENDDPRCAP: Stopping Capture (OS/400)” on page 400 for command syntax and parameter descriptions.)

**Recommendation:** Reorganize the following dynamic control tables once a week:

- CD tables
- IBMSNAP\_ALERTS
- IBMSNAP\_APPLYTRACE
- IBMSNAP\_APPLYTRAIL
- IBMSNAP\_CAPMON
- IBMSNAP\_CAPTRACE

- IBMSNAP\_MONTRAIL
- IBMSNAP\_MONTRACE
- IBMSNAP\_UOW

You do not need to run any utilities that reclaim unused space or generate frequently updated optimizer statistics on the static control tables:

- Apply enqueue (IBMSNAP\_APPENQ)
- Capture enqueue (IBMSNAP\_CAPENQ) (UNIX, Windows, z/OS)
- Capture parameters (IBMSNAP\_CAPPARMS)
- Capture schemas (IBMSNAP\_CAPSCHEMAS)
- Prune lock (IBMSNAP\_PRUNE\_LOCK)
- Prune set (IBMSNAP\_PRUNE\_SET)
- Pruning control (IBMSNAP\_PRUNCNTL)
- Register (IBMSNAP\_REGISTER)
- Register extension (IBMSNAP\_REG\_EXT) (OS/400 only)
- Register synchronization (IBMSNAP\_REG\_SYNCH)
- Replication Alert Monitor conditions (IBMSNAP\_CONDITIONS)
- Replication Alert Monitor contacts (IBMSNAP\_CONTACTS)
- Replication Alert Monitor contact groups (IBMSNAP\_CONTACTGRP)
- Replication Alert Monitor enqueue (IBMSNAP\_MONENQ)
- Replication Alert Monitor groups (IBMSNAP\_GROUPS)
- Replication Alert Monitor servers (IBMSNAP\_MONSERVERS)
- Restart (IBMSNAP\_RESTART)
- Sequencing (IBMSNAP\_SEQTABLE)
- Subscription columns (IBMSNAP\_SUBS\_COLS)
- Subscription members (IBMSNAP\_SUBS\_MEMBR)
- Subscription sets (IBMSNAP\_SUBS\_SET)
- Subscription statements (IBMSNAP\_SUBS\_STMTS)

## Pruning your control tables

You should regularly prune your replication control tables to remove obsolete data and to improve system performance. This section discusses the different methods that you can use to prune your control tables and how these methods affect the performance of your system.

### Pruning dynamic control tables maintained by the Capture programs

You should monitor the growth of and consider the various pruning methods available for the following dynamic control tables:

- CD tables
- IBMSNAP\_UOW (UOW)

- IBMSNAP\_CAPMON
- IBMSNAP\_CAPTRACE
- IBMSNAP\_SIGNAL
- IBMSNAP\_AUTHTKN (OS/400 only)

You can set your Capture programs to prune these tables automatically at regular intervals. Or you can prune on demand by launching the pruning process once; the Capture program does not prune again until you enter the next prune command.

**Recommendation:** Consider the use of automatic pruning to manage the growth of these control tables. Automatic pruning minimizes storage costs, increases the efficiency of the your Apply programs, and generally reduces the risk of system failure due to storage overflow by regularly removing obsolete data from these tables. To invoke automatic pruning:

- Set the Capture program **autoprune** parameter to y (UNIX, Windows, z/OS).
- Use the **CLNUPITV(\*IMMED)** or **CLNUPITV(\*DELAYED)** Capture program parameter setting (OS/400).

With autopruning, you set the **prune\_interval** operational parameter (on UNIX, Windows, and z/OS) or the **RETAIN** parameter (on OS/400) to specify how frequently the automatic pruning process occurs.

### Procedure:

Use one of the following methods to initiate pruning:

#### Replication Center

Use the Prune Capture Control Tables window to prune the tables once. See the Replication Center help for details.

#### **asncap system command with autoprune=y (UNIX, Windows, z/OS)**

Use this command to start a Capture program with automatic pruning. See “asncap: Starting Capture (UNIX, Windows, z/OS)” on page 316 for command syntax and parameter descriptions.

#### **asnccmd system command with chgparms autoprune=y (UNIX, Windows, z/OS)**

Use this command to enable automatic pruning in a running Capture program. See “asnccmd: Operating Capture (UNIX, Windows, z/OS)” on page 322 for command syntax and parameter descriptions.

#### **asnccmd system command with the prune parameter (UNIX, Windows, z/OS)**

Use this command to initiate pruning once from a running Capture program. See “asnccmd: Operating Capture (UNIX, Windows, z/OS)” on page 322 for command syntax and parameter descriptions.



**STRDPRCAP CLNUPITV(\*IMMED) or STRDPRCAP CLNUPITV(\*DELAYED) system commands (OS/400)**

Use these commands to prune old rows at specified intervals after starting a Capture program. See “STRDPRCAP: Starting Capture (OS/400)” on page 436 for parameter descriptions and command syntax.

**OVRDPRCAPA PRUNE(\*IMMED) or OVRDPRCAPA PRUNE(\*DELAYED) system command (OS/400)**

Use this command to change the way that a running Capture program prunes these control tables. See “OVRDPRCAPA: Overriding DPR capture attributes (OS/400)” on page 414 for command syntax and parameter descriptions.

**Pruning the CD and UOW tables:** During each pruning cycle, whether invoked automatically or on demand, the Capture program prunes the CD and UOW tables based on the progress reported by the Apply programs. Progress is indicated by the SYNCHPOINT column value in the prune set (IBMSNAP\_PRUNE\_SET) table. This *normal* pruning is based on the minimum synchpoint value over all Apply programs that subscribe to each CD table and on the minimum overall synchpoint value for the UOW table.

Normal pruning, however, does not prune the CD and UOW tables effectively if the associated subscriptions sets run very infrequently. Keep pruning effectiveness in mind when deciding how often to run the associated Apply programs, when stopping these Apply programs, and when deactivating the subscription sets for more than a brief period of time.

If you run your subscription sets very infrequently or stop your Apply programs, your CD and UOW tables can grow very large and become eligible for retention limit pruning. The retention limit is an operational parameter of the Capture program, with a default value of one week. It determines how long old data remains in the tables before becoming eligible for retention limit pruning.

If the normal pruning process is inhibited due to deactivated or infrequently run subscription sets, data can remain in the table for long periods of time. If this data becomes older than the current DB2 timestamp minus the retention limit value, the retention limit pruning process prunes this data from the tables.

Try to avoid conditions that require retention limit pruning, because the accumulation of old data can lead to storage overflows and performance degradation. See “Deactivating subscription sets” on page 212 for more information.

**Recommendation:** Run your Apply programs at least once per day for all of your subscription sets.

If the source server is supplying changed data to a variety of target systems, each with very different requirements and some with infrequently running Apply programs for few registered sources, consider the use of multiple Capture programs. You can use multiple Capture programs and manage the various processing requirements with different Capture schemas, using one Capture schema to isolate those tables that are infrequently pruned due to specific subscription-set timing requirements and using another Capture schema for the remaining source tables.

**Pruning the Capture monitor and Capture trace tables:** During each pruning cycle, the Capture program prunes the Capture monitor (IBMSNAP\_CAPMON) and the Capture trace (IBMSNAP\_CAPTRACE) tables based on the values of the following operational parameters of the Capture program:

- The **monitor\_limit** parameter (on UNIX, Windows, z/OS) and the **MONLMT** parameter (on OS/400) that indicate how long rows remain in the IBMSNAP\_CAPMON table
- The **trace\_limit** parameter (on UNIX, Windows, z/OS) and the **TRCLMT** parameter (on OS/400) that indicate how long rows remain in the IBMSNAP\_CAPTRACE table

Both the monitor limit and the trace limit parameters have a default value of one week. You can change these values depending on how long you need to preserve the historical Capture latency and throughput information in the IBMSNAP\_CAPMON table and the auditing and troubleshooting data from the IBMSNAP\_CAPTRACE table.

**Pruning the signal table:** The signal (IBMSNAP\_SIGNAL) table is also pruned during each pruning cycle. A signal row is eligible for pruning if the SIGNAL\_STATE column value is equal to C. A value of C indicates that the signal information is complete and is no longer required by the Capture program or for any user processing and is eligible for pruning. A signal row with a SIGNAL\_TIME column value that is older than the current DB2 timestamp minus the retention limit parameter value is eligible for retention limit pruning.

### **Pruning other dynamic control tables**

The Capture program performs pruning operations for only the tables that it maintains. The Apply program maintains consistent-change data (CCD) tables; therefore, the Capture program does not automatically prune these tables. Some types of CCD tables do not require pruning. Complete condensed CCD tables are updated in place.

The only records that you might want to remove from complete condensed CCD tables are those with an IBMSNAP\_OPERATION column value of D (Delete) that have already been replicated to the dependent target tables. Non-condensed CCD tables contain historical data and can grow very large. Because you should preserve this data for auditing purposes, you should not perform pruning operations on non-condensed CCD tables.

You should, however, consider pruning your internal CCD tables. These tables can grow quickly if there is heavy update activity on your system. Only the most recent changes are fetched from internal CCD tables, so you do not need to retain the older rows.

To enable pruning for internal CCD tables, consider adding after-SQL statements to associated subscription sets to prune change data that has already been applied to all dependent targets. Alternatively, you can also add the necessary SQL DELETE statements to your automatic scheduling facilities to delete rows from these tables.

You should also manually prune the Apply trail (IBMSNAP\_APPLYTRAIL) and Apply trace (IBMSNAP\_APPLYTRACE) tables. If you define and use multiple subscription sets with frequently run Apply programs, the IBMSNAP\_APPLYTRAIL table grows rapidly and requires frequent pruning. The best way to manage the growth of these tables is to add an after-SQL statement or procedure call to one of your subscription sets. Alternatively, you can add an SQL DELETE statement to your automatic scheduling facilities.

## **Preventing replication failures and recovering from errors**

This section describes methods to prevent and recover from replication failures that can affect your control tables and replication data:

- Preventing cold starts of the Capture program
- Recovering from I/O errors and connectivity failures on your control tables
- Retrieving lost source data

### **Preventing cold starts of the Capture program**

You should perform a cold start of the Capture program only if you are starting the program for the first time or you need to refresh your control and target tables. If you cold start the Capture program, all of the target tables in your replication environment are refreshed.

When a Capture program starts with the warmns, warmsa, or warmsi option on UNIX, Windows or z/OS, the program attempts to retrieve log records based on the restart point in the restart (IBMSNAP\_RESTART) table. If the Capture program cannot find the log, the Capture warm start fails. If you started the Capture program using the warmns or warmsi option, the restart process terminates and issues an error message. If you started the Capture

program using the warmsa option, the restart process stops and the Capture program performs a cold start, deleting all records in the CD and UOW tables.

To prevent a cold start of the Capture program, consider the following recommendations:

- On UNIX, Windows, and z/OS operating systems, specify the warmns or warmsi startmode instead of warmsa to restart a Capture program whenever possible. The warmns and warmsi options prevent an automatic cold start of the Capture program if the restart process fails. See “asncap: Starting Capture (UNIX, Windows, z/OS)” on page 316 for more information.
- On OS/400 operating systems, start the capture program with the **RESTART(\*YES)** parameter. The Capture program continues processing from the point where it was when it ended previously. See “STRDPRCAP: Starting Capture (OS/400)” on page 436 for more information.
- Use the Replication Alert Monitor or other mechanism to check the status of the historical data from your Capture programs. You can then use this information to verify that the Capture programs are always running if DB2 is active. See Chapter 11, “Monitoring replication” on page 161 for more information.
- Make sure that you retain sufficient DB2 log data or journal receivers on your system and that this data is available to DB2 replication. See “Maintaining and retaining source logs and journal receivers” on page 226 for information regarding log retention.

### **Recovering from I/O errors and connectivity failures on your control tables**

If you experience an I/O error or connectivity failure on any control table, use a standard DB2 recovery procedure to forward recover the table; the table will not lose any data.

If the Capture program detects an I/O error or connectivity failure, the program issues an appropriate error message and shuts down. After you correct the error, you can restart the Capture program from the point of failure.

The Apply program shuts down if it detects catastrophic errors on the control tables. If the Apply program detects errors on target tables or errors with network connectivity, the program writes the error to the Apply trail (IBMSNAP\_APPLYTRAIL) table and then continues processing.

### Retrieving lost source data

If a source table is forward recovered to the point of failure, DB2 replication proceeds normally. After the table is recovered, the Capture program continues collecting data changes for the table.

However, the Capture and Apply programs do not detect a point-in-time recovery of a read-only target table. If you recover a source table, the Apply program might have replicated changes to the target tables that no longer exist at the source, leaving inconsistencies between your source tables and target tables if you cannot take the target tables back to the same logical point in time.

This scenario becomes even more complex when there are multiple levels of replication. You must either develop a mechanism that provides matching recovery points among the various levels or use a full refresh as your recovery method of choice.

See “Coordinating replication events with database application events” on page 214 for more information about setting distributed recovery points.

---

## Maintaining your target tables

Maintain the tables on the target server in the same way that you maintain other tables on your database system. Use your current backup and maintenance routines on these target tables, whether your target tables are existing database tables or tables that you specified to be automatically generated by DB2 replication.

**Important:** Deactivate your Apply programs before taking a target table offline to run any utility.

### Related concepts:

- Chapter 22, “How the DB2 replication components communicate” on page 465

### Related tasks:

- Chapter 1, “Planning for replication” on page 3
- Chapter 2, “Setting up for replication” on page 15



---

## Part 2. Replication Center

This part of the book contains the following chapters:

Chapter 14, “Using the DB2 Replication Center” on page 243 describes the Replication Center.

Chapter 15, “Basic data replication scenario: DB2 for Windows” on page 269 describes how to use the Replication Center to perform a simple replication scenario on sample data.





---

## Chapter 14. Using the DB2 Replication Center

The DB2<sup>®</sup> Replication Center is a user interface tool that you can use to set up and administer your replication environment and to run the Capture, Apply, and Replication Alert Monitor programs. You can use the Replication Center to perform such administration tasks as:

- Create replication control tables
- Register replication sources
- Create subscription sets and add subscription-set members to the set
- Operate the Capture program
- Operate the Apply program
- Monitor the replication process

The Replication Center also has a launchpad that allows you to perform the basic functions needed to set up a DB2 replication environment. The launchpad shows you graphically how the different steps are related to one another.

You can use the Replication Center to set up DB2-to-DB2 replication environments or replication between DB2 and non-DB2 relational databases. The DB2 Replication Center is part of the DB2 Control Center set of tools. See the online help for detailed task information for the Replication Center.

This chapter helps you perform the following tasks:

### Using the Replication Center

- See “Prerequisites for the DB2 Replication Center” on page 245. The prerequisites for the Replication Center are very similar to the prerequisites for the DB2 Control Center.
- See “Starting the DB2 Replication Center” on page 245. You can start the Replication Center several ways.
- See “Using the Replication Center launchpad” on page 246. Using the launchpad is optional, but can be very helpful for first-time users.

### Setting up the Replication Center

- See “Managing user IDs and passwords for the Replication Center” on page 247. You can maintain the passwords that the Replication Center uses to connect to databases and log on to systems.

- See “Creating replication profiles” on page 248. Creating profiles is optional, but can be very helpful if you manage a large replication environment.
- See “Creating replication control tables” on page 251. You must create control tables in each database that will act as a replication control server.
- See “Adding servers to the Replication Center” on page 254. Servers are added to the Replication Center automatically when you create replication control tables. You can customize your view of your replication environment by adding only those servers that you want to administer.
- See “Enabling a database for change capture (UNIX and Windows)” on page 255. You must enable each Capture control server on UNIX<sup>®</sup> and Windows<sup>®</sup> systems for change capture and initiate a database backup.

### **Defining the replication environment**

- See “Registering sources” on page 256. You can register tables or views as replication sources.
- See “Creating subscription sets” on page 257. You can create empty sets and add subscription-set members to them at any time, or you can create the subscription-set members as you create the subscription set.

### **Maintaining your replication environment**

- See “Activating or deactivating subscription sets” on page 261. You can temporarily or permanently deactivate or activate any subscription set.
- See “Promoting replication objects” on page 262. You can promote table registrations and subscription sets from a test environment to a production environment.
- See “Forcing a full refresh of target tables” on page 263. You can control when the Apply program performs a full refresh for a subscription set
- See “Removing or deleting replication definitions” on page 264. You can remove replication objects from the Replication Center and you can delete replication definitions from a replication control server.

### **Operating your replication environment**

- See “Operating the Capture program” on page 265. You can start and stop the Capture program on any server in your network. You can also perform many other operational tasks for the Capture program.

- See “Operating the Apply program” on page 266. You can start and stop the Apply program on any server in your network. You can also perform many other operational tasks for the Apply program.
- See “Operating the Replication Alert Monitor” on page 266. You can define alert conditions for monitoring replication activity.

---

## Prerequisites for the DB2 Replication Center

Your system must have the correct Java™ Runtime Environment (JRE) installed to run the Replication Center. When you install DB2, you have the option to install the JRE. If you choose not to install the JRE, you must ensure that your system has Version 1.3 of either the Java 2 Runtime Environment or the Java 2 Software Development Kit.

If you will use the Replication Center to operate the Capture, Apply, or Replication Alert Monitor programs on remote systems, ensure that the DB2 Administration Server (DAS) is running on the local system that is running the Replication Center and on each of the remote DB2 systems that will run the Capture or Apply programs.

---

## Starting the DB2 Replication Center

The Replication Center is installed as part of a typical DB2 installation for UNIX or Windows operating environments. If you perform a custom installation, you must select the General Administration Tools component to install the Replication Center.

To start the Replication Center, enter the **db2rc** command in a command window.

On Windows systems, you can also start the Replication Center by using the Windows **Start** menu:

1. Click **Start**.
2. Select **Programs**.
3. Select **IBM DB2**.
4. Select **General Administration Tools**.
5. Click **Replication Center**.

If you already have the DB2 Control Center running, you can start the Replication Center by selecting **Replication Center** from the **Tools** menu or by clicking the icon for the Replication Center.

Figure 7 on page 246 shows the Replication Center.

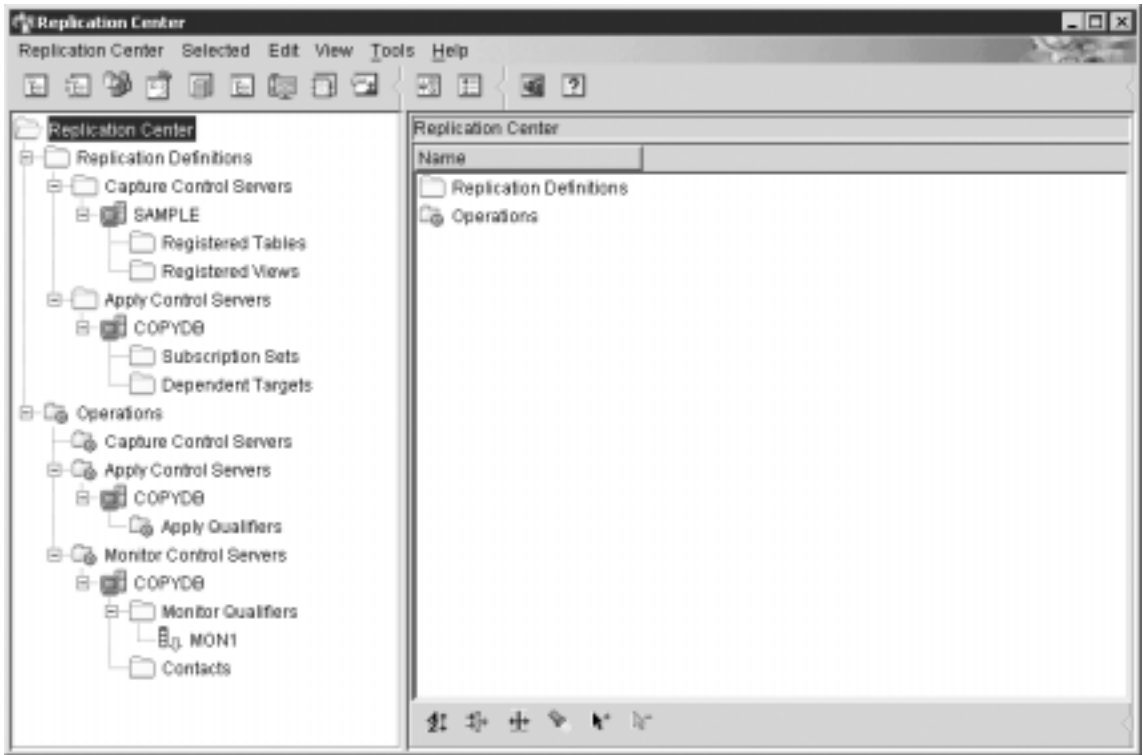


Figure 7. The DB2 Replication Center

## Using the Replication Center launchpad

When you first start the Replication Center, you see the Replication Center launchpad. You can use the launchpad to perform the basic functions needed to set up a DB2 replication environment. It also shows you graphically how the different steps are related to one another.

The launchpad gives you central access to the most common functions in the Replication Center, but you can also access these functions from the Replication Center navigator. You can use the Replication Center navigator to view or manipulate anything that you create using the launchpad. Many other, more advanced, functions are available in the Replication Center that are not accessible from the launchpad.

From the launchpad, you can perform the following tasks:

- Create the Capture control tables

This option opens the Create Capture Control Tables window, from which you can create the necessary replication control tables in a specific database for the Capture program.

- Register a source table

This option opens the Register Tables window, from which you can define registration information (source columns, CD table information, and so on) for each source table that you want to register.

- Create the Apply control tables

This option opens the Create Apply Control Tables window, from which you can create the necessary replication control tables in a specific database for the Apply program.

- Create a subscription set

This option opens the Create Subscription Set window, from which you can define subscription set information: Capture, target, and Apply control servers, source-target mapping, properties for each subscription set member, schedule for the set, and SQL statements for the set.

- Start the Capture program

This option opens the Start Capture window, from which you can start the Capture program and specify startup parameters for it.

- Start the Apply program

This option opens the Start Apply window, from which you can start the Apply program and specify startup parameters for it.

**Recommendation:** The launchpad does not require that you perform these tasks in sequence, but if you are new to DB2 replication, follow the sequence that is presented on the launchpad. You can also skip or repeat steps in the launchpad if the necessary replication or database objects already exist. Use the launchpad or the Replication Center navigator to create the necessary replication and database objects.

You can use the launchpad at any time by selecting **Launchpad** from the **Replication Center** menu, or by right-clicking the Replication Center folder in the navigator and selecting **Start Launchpad**.

---

## Managing user IDs and passwords for the Replication Center

The Replication Center must be able to connect to many database servers: source servers, Capture control servers, Apply control servers, Monitor control servers, and target servers. The Replication Center must also be able to connect to each system that runs the Capture program, the Apply program, or the Replication Alert Monitor. For all remote databases and systems, you need a valid user ID and password to connect to each database or to log on to each system. The Replication Center allows you to specify each user ID and

password once, so that you are not prompted every time the Replication Center attempts to connect to a remote database or log on to each remote system.

By default, the Replication Center saves the user ID and password information in its metadata file. Because passwords are not encrypted in this file, you can specify that the Replication Center should not save passwords in this file but keep them only in memory. When you change a password in DB2, you must also change the password in the Replication Center so that they both use the same password. The Replication Center does not share its password information with the Apply program, Replication Analyzer, or Replication Alert Monitor.

To manage user IDs and passwords for the Replication Center:

1. Right-click the **Replication Center** folder.
2. Select **Manage Passwords for Replication Center**.

In the Manage Passwords for Replication Center window, you can perform any of the following tasks:

- Add a new user ID and password pair for a database alias or a system name
- Change a password for an existing user ID
- Remove a user ID
- Test the connection to a selected database alias or the logon for the selected system using a user ID and password

---

## Creating replication profiles

As part of replication setup, you create replication control tables, often in more than one database; you register many source tables and views, all of which have CD tables; and you define many target tables as part of creating subscription sets. For each of these replication objects, you probably have specific naming conventions and common attributes (for example, perhaps all of the table spaces for your CD tables use the same page size). The Replication Center allows you to create profiles that reflect these naming conventions and common definitions, rather than having to specify these common definitions every time you create replication objects. You can create profiles for the following replication objects:

- Replication control tables
- Replication source objects (CD tables)
- Replication target objects

In each object profile, you specify naming conventions for database objects such as CD tables, indexes, and table spaces, and you specify common

attributes for these objects, such as page sizes and buffer pools. The values that you specify in each profile become the default values displayed in the Create Control Tables window, the Register Tables window, the Register Views window, or the Create Subscription Set window. You can override these default values when you create specific replication objects, or you can accept the values that you defined in your profile by clicking **OK**.

## Creating control-table profiles

For each replication control table (for example, the register table, IBMSNAP\_REGISTER), you can define table space information and index information in the profile. By default, the Replication Center groups the replication control tables together into table spaces for optimal performance. For these table spaces, you can define a naming convention that the Replication Center uses when it creates the table space, or you can specify a table space that already exists. You can also define other operating-system specific table space information for the control tables. Most of the control tables also require one or more indexes. You can define a naming convention that the Replication Center uses when it creates these indexes, or you can specify an index that already exists.

You can define a unique control-table profile for each type of operating system that DB2 supports. You can also define profiles for each type of non-DB2 database that DB2 replication supports. The Replication Center does not provide a control-table profile for OS/400® systems because the replication control tables are created when you install DB2 DataPropagator™ for iSeries.

To create control-table profiles:

1. Right-click the **Replication Definitions** folder.
2. Select **Manage Control Table Profiles**.

In the Manage Control Table Profiles window:

- a. Select the operating-system or non-DB2 database environment for which you are creating the profile.
- b. Select a replication control table from the list.
- c. Define the table space and index characteristics for the selected control table.

Be sure to select every control table that creates a separate table space and define its characteristics.

- d. When you have defined all of the control tables for a particular operating-system platform or non-DB2 relational database system, click **Apply**. Click **Close** to close the Manage Control Table Profiles window.

## Creating source-object profiles

For each DB2 replication source object (table or view), the Capture program requires a CD table. When you register a source object, you specify the name and characteristics for both the CD table and the index for the CD table. By creating a profile for source objects, you can define common characteristics for all sources that you register from a particular source database. Using these common characteristics, you can register many tables or views as part of a single action.

You can define a naming convention that the Replication Center uses when it creates the CD table, the table space for the CD table, and the index for the CD table. You can also define truncation rules for each of these objects if the name should exceed the operating-system-specific length limit (for example, 128 characters for UNIX and Windows databases) after the Replication Center applies the prefix and suffix that you specify. For example, you can create a profile that names your CD tables “CD\_*sourcetable*name” (where *sourcetable*name varies for each registered source table) and places them in a table space named “CD\_repltablespace”.

You must add a Capture control server to the Replication Center before you can create a source-object profile for that server.

To create source-object profiles, use the Manage Source Object Profiles window. You can open this window in two ways:

- Open the window from the **Replication Definitions** folder:
  1. Right-click the **Replication Definitions** folder.
  2. Select **Manage Source Object Profiles**.  
In the Manage Source Object Profiles window, select the source server for which you are creating the profile.
- Open the window from a source server:
  1. Expand the **Replication Definitions** folder.
  2. Expand the **Capture Control Servers** folder.
  3. Right-click a source server and select **Manage Source Object Profiles**.  
In this case, in the Manage Source Object Profiles window, you don't need to select the source server for which you are creating the profile.

In the Manage Source Object Profiles window, define the characteristics for the CD table, the table space for the CD table, and the index for the CD table. You can also select the truncation rules for each of these objects.

## Creating target-object profiles

When you create a subscription-set member, you define a replication mapping between a source object (table, view, or nickname) and a target table. If the target table does not already exist, you specify the name and characteristics



for both the target table and the index for the target table. By creating a profile for target objects, you can define common characteristics for all target tables in a particular target database.

You can define a naming convention that the Replication Center uses when it creates the target table, the table space for the target table, and index for the target table. You can also define truncation rules for each of these objects if the name should exceed the operating-system-specific length limit (for example, 128 characters for UNIX and Windows databases) after the Replication Center applies the prefix and suffix that you specify. For example, you can create a profile that names your target tables “TG\_*sourcetable*name” (where *sourcetable*name varies for each registered source table) and places them in a table space named “TS\_*targettable*name” (where *targettable*name varies for each target table).

You must catalog a target server in the local DB2 database before you can create a target-object profile for it. However, you do not need to add the target server to the Replication Center as a Capture control server, an Apply control server, or a Monitor control server.

To create target-object profiles:

1. Right-click the **Replication Definitions** folder.
2. Select **Manage Target Object Profiles**.
3. In the Select Server window, select the database server for which you are creating a target-table profile and click **OK**.

In the Manage Target Object Profiles window, select the target server for which you are creating the profile. Define the characteristics for the target table, the table space for the target table, and the index for the target table. You can also select the truncation rules for each of these objects.

---

## Creating replication control tables

The replication control tables store all of the information about the setup for your replication environment, and they store operational information that the Capture and Apply programs use during replication. You must create replication control tables in a database before you can add that database server to the Replication Center. When you create replication control tables in a particular database, the Replication Center automatically adds that database server to the Replication Center navigator.

You cannot use the Replication Center to create replication control tables for OS/400 systems because the replication control tables are created when you install DB2 DataPropagator for iSeries. If you want to recreate the control tables, or create the control tables using an alternate Capture schema, use the OS/400 **CRTDPRTBL** command.

A Capture control server can also act as an Apply control server if you create all of the replication control tables in the same database. Likewise, a Capture or Apply control server can also act as a Monitor control server.

## Creating Capture control tables

You can create control tables for a Capture control server in two ways:

- Open the Create Control Tables – Quick window:
  1. Expand the **Replication Definitions** folder.
  2. Right-click the **Capture Control Servers** folder and select **Create Capture Control Tables → Quick**.
  3. In the Select a Server window, select the server in which you want to create Capture control tables and click **OK**.

The Create Control Tables – Quick window asks you some simple questions about your replication environment, and based on your answers, the Replication Center will create the replication control tables in specific table spaces of the appropriate sizes, and group the control tables in these table spaces for optimal performance.

- Open the Create Capture Control Tables window:
  1. Expand the **Replication Definitions** folder.
  2. Right-click the **Capture Control Servers** folder and select **Create Capture Control Tables → Custom**.
  3. In the Select a Server window, select the server in which you want to create Capture control tables and click **OK**. This server is where you will run the Capture program. If the database is a federated database that acts as a gateway for non-DB2 relational sources, select the federated database and retrieve the server mappings from that database to display a list of non-DB2 relational servers that are defined for that federated database.

In the Create Capture Control Tables window, define the characteristics for each control table:

- a. Select a replication control table from the list.
- b. Define the table space and index characteristics for the selected control table.

Be sure to select every control table that creates a separate table space and define the table space and index characteristics for that control table.

**Tip:** If you created a control-table profile for the operating-system platform of the selected database, you can accept the settings from your profile or override them.

You can specify a unique schema name for the Capture control tables; the default is ASN. A separate schema name is necessary if you plan to run more than one instance of the Capture program for the selected database.

- c. When you have defined all of the control tables, click **OK**.

## Creating Apply control tables

You can create control tables for an Apply control server in two ways:

- Open the Create Control Tables – Quick window:
  1. Expand the **Replication Definitions** folder.
  2. Right-click the **Apply Control Servers** folder and select **Create Capture Control Tables → Quick**.
  3. In the Select a Server window, select the server in which you want to create Apply control tables and click **OK**.

The Create Control Tables – Quick window asks you some simple questions about your replication environment, and based on your answers, the Replication Center will create the replication control tables in specific table spaces of the appropriate sizes, and group the control tables in these table spaces for optimal performance.

- Open the Create Apply Control Tables window:
  1. Expand the **Replication Definitions** folder.
  2. Right-click the **Apply Control Servers** folder and select **Create Capture Control Tables → Custom**.
  3. In the Select a Server window, select the server in which you want to create Apply control tables and click **OK**.

In the Create Apply Control Tables window, define the characteristics for each control table:

- a. Select a replication control table from the list.
- b. Define the table space and index characteristics for the selected control table.

Be sure to select every control table that creates a separate table space and define the table space and index characteristics for that control table.

**Tip:** If you created a control-table profile for the operating-system platform of the selected database, you can accept the settings from your profile or override them.

- c. When you have defined all of the control tables, click **OK**.

## Creating Monitor control tables

You can create Monitor control tables in UNIX, Windows, or z/OS™ databases. You should not create Monitor control tables in OS/400 databases.

or non-DB2 relational databases. You can use a Monitor control server to monitor replication activity for any DB2 database in your replication network, including OS/400 databases.

To create control tables for a Monitor control server:

1. Expand the **Operations** folder.
2. Right-click the **Monitor Control Servers** folder and select **Create Monitor Control Tables**.
3. In the Select a Server window, select the server in which you want to create Monitor control tables and click **OK**.

In the Create Monitor Control Tables window, define the characteristics for each control table:

- a. Select a replication control table from the list.
- b. Define the table space and index characteristics for the selected control table.

Be sure to select every control table that creates a separate table space and define the table space and index characteristics for that control table.

- c. When you have defined all of the control tables, click **OK**.

---

## Adding servers to the Replication Center

When you create replication control tables in a particular database, the Replication Center automatically adds that database server to the Replication Center navigator. You can also add or remove a database server from the Replication Center navigator without affecting any of the replication objects that you created in that database and without affecting the Capture program, the Capture triggers, the Apply program, or the Replication Alert Monitor that might be running on that server. The Replication Center does not automatically list all of your locally cataloged databases for the following reasons:

- The Replication Center shows only valid replication objects. If a locally cataloged database does not contain replication control tables, Replication Center does not display that database in the navigator.
- If your replication environment restricts the authority for creating replication control tables to an administrator, you can still allow other people to manage replication objects (for example, registered sources or subscription sets) in databases of interest to them.
- Even if everyone in your replication team has the same authority, each person might want to focus only on certain replication servers. Each person can add just those database servers that he or she wants to administer, even if your replication environment includes more than the Replication Center shows.

**Important:** Before you can add a database server to the Replication Center, you must first catalog the server in the local DB2 database and ensure that replication control tables exist in the database.

You can add the following servers to the Replication Center:

- Capture control server

To add a Capture control server to the Replication Center, right-click the **Capture Control Servers** folder and select **Add**. In the Add Capture Control Servers window, select one or more database servers from the list.

You can also add non-DB2 relational servers to the Replication Center as Capture servers by right-clicking a particular federated database displayed in the Add Capture Control Servers window and selecting **Retrieve non-DB2 Server(s)**. The Replication Center adds the non-DB2 relational server defined for that federated database to the table.

- Apply control server

To add an Apply control server to the Replication Center, right-click the **Apply Control Servers** folder and select **Add**. In the Add Apply Control Servers window, select one or more database servers from the list.

- Monitor control server

To add a Monitor control server to the Replication Center, right-click the **Monitor Control Servers** folder and select **Add**. In the Add Monitor Control Servers window, select one or more database servers from the list.

For each server that you add, you must enter a valid user ID and password that the Replication Center can use to connect to the selected database. The Replication Center uses the saved password information for each server, if it has any.

---

## Enabling a database for change capture (UNIX and Windows)

The default logging for a DB2 database on UNIX or Windows systems is circular logging, which uses a fixed-size file that is reused when the log fills. Replication requires archival logging, which uses one or more log files that grow indefinitely and are never reused (of course, you can use DB2 utilities to manage an archive log to ensure that it doesn't fill all your disk space).

To enable archival logging for DB2 databases:

1. Expand the **Replication Definitions** folder.
2. Expand the **Capture Control Servers** folder.
3. Right-click the database for which you want to enable archival logging, and select **Enable Database for Replication**.

4. Click **OK** on the Enable Database for Replication window to change the database configuration (to set LOGRETAIN to RECOVERY) and to initiate a database backup.

If you also want to use an exit routine to manage archived logs, you must manually set the USEREXIT database-configuration parameter.

You can also select several databases in the contents pane for the **Capture Control Servers** folder and enable archival logging for all selected databases at the same time.

You do not need to enable archival logging for databases on other operating-system platforms because the default logging for those environments is archival. You also do not need to enable archival logging for non-DB2 relational databases because the Capture triggers do not depend on the database logs.

---

## Registering sources

To register one or more tables for replication:

1. Expand the **Capture Control Servers** folder.
2. Expand the database server that contains the source tables that you want to register.
3. Right-click the **Registered Tables** folder and select **Register Tables**. The Add Registrable Tables window opens.

Because the database could contain many hundreds of tables, you can prefilter the list of tables so that the Register Tables window shows only those tables that you are interested in.

4. From the Add Registrable Tables window, specify the search criteria, if any, and click **Retrieve**. If you want to include all tables, click **Retrieve All**.
5. Select one or more tables from the filtered list that you want to register as a replication source and click **OK**. The Register Tables window remains open.
6. From the **Selected tables** list, select the first table that you want to register as a replication source. You can define the following information for a replication source:
  - The row-capture rule that specifies when the Capture program writes a row to the CD table (or when the Capture triggers write a row to the consistent-change data (CCD) table).
  - The specific columns that you want to make available for replication, including before-image and after-image columns.

Any column that you do not register will not be available for any subscription set.

- The prefix to use for registered before-image columns to associate them with the after-image columns.
- Whether you want to allow the Apply program to refresh target tables based on this source table.
- Whether to capture changes as delete and insert pairs (useful for changes to partitioning keys).
- Whether changes are recaptured in dependent replicas in an update-anywhere scenario.
- The level of conflict detection for update-anywhere scenarios.  
For peer-to-peer scenarios, you must select **No detection**.

For more information about these options, see the online help for the Replication Center.

For each registered source table, you also specify information about the CD table and the index for the CD table. If you created a source-object profile for this database server, you can accept the defaults that you defined in the profile, or you can override them.

To register a view, right-click the **Registered Views** folder and select **Register Views**. The view must already exist before you can register it as a replication source. If it does not exist, click **Create View** in the Register Views window. In the Create View window, specify the view name and the SQL statement that defines the view. You can click **SQL Assist** to use the SQL Assist window to create the SQL statement that defines the view.

You can register an OS/400 table that is journaled remotely in the same way as you register any table, except that you must specify the source-table name as well as the journal library and journal-receiver name. However, you do not need to specify the journal library and journal name if they are the same as the journal library and journal name used by the source table or file.

You can register a nickname in the same way as you register a table, except that you must specify the nickname for the table (stored in DB2) instead of the actual table name (stored in the non-DB2 database).

---

## Creating subscription sets

After you register one or more source tables, nicknames, or views, you need to subscribe to those sources; that is, to create a subscription set and add members to the set. You can create an empty subscription set and add members to it later, or you can add all the members while you create the subscription set.

To create a subscription set:

1. Expand the **Capture Control Servers** folder.
2. Expand the database server that contains the source tables for which you want to create subscription sets.
3. Click the **Registered Tables** folder.
4. In the contents pane for the **Registered Tables** folder, right-click a source table and select **Create Subscription Set**. The Create Subscription Set window opens.

As an alternative, you can also create a subscription set using the following steps:

1. Expand the **Apply Control Servers** folder.
2. Expand a specific Apply control server.
3. Right-click the **Subscription Sets** folder, and select **Create**. The Create Subscription Set window opens.

Creating a subscription set includes four major subtasks:

- “Defining the information for the subscription set”.
- “Mapping sources to targets” on page 259.
- “Scheduling the subscription set” on page 260.
- “Adding SQL statements or stored procedures to the subscription set” on page 260.

After you create the subscription set, you can edit the subscription set, add or remove subscription-set members, add or remove statements or procedures, activate the subscription set, force a full refresh of its members, or promote it to another database.

For more information about creating subscription sets, see the online help for the Replication Center.

## Defining the information for the subscription set

In the Create Subscription Set window, you can define the following information for the subscription set:

- The alias for the Apply control server
- The subscription-set name
- The Apply qualifier
- The alias for the Capture control server
- The Capture schema that identifies the set of Capture control tables that define the registered sources for the subscription set
- The alias for the target server
- Whether the subscription set should be active as soon as it is created



By default, a new subscription set is deactivated it as soon as it is created. You can choose to make it eligible to be processed by the Apply program immediately, or you can choose to activate it for just one Apply cycle.

- The subscription-set processing properties

## Mapping sources to targets

After you define the subscription-set information, you can map the source tables and views to the target tables. On the Source-to-Target Mapping page of the Create Subscription Set window:

1. Click **Add** to display the Add Registered Sources window. From this window, you can filter the list of registered sources for the selected source database.
2. Select one or more tables from the filtered list that you want to add as the source for the subscription-set member and click **OK**. The Create Subscription Set window remains open at the Source-to-Target Mapping page.
3. In the table on the Source-to-Target Mapping page, select the target schema, name, or target type to change any of these values for target tables that do not already exist. After you choose a source, the target schema and target name are automatically generated based on the target-object profile for the selected target server, if one exists.
4. In the table on the Source-to-Target Mapping page, select a source-target pair in the table and click **Details** to display the Member Properties window. From this window, you can specify the exact mapping between a source table and a target table, including:
  - Selecting to which source columns the target will subscribe
  - Mapping source columns to target columns, including creating calculated columns
  - Specifying the index for the target table
  - Optionally, filtering the source rows with a WHERE clause, so that the target table includes only a subset of the source data
  - For UNIX, Windows, and z/OS systems, specifying the table space for the target table

For replica target types you also specify the replica definition (row-capture rule, whether to recapture changes, and how to handle updates), the CD table for the replica table, and the index for the CD table.

For CCD tables you also specify the properties of the CCD table, including whether it is complete or noncomplete, condensed or noncondensed, and whether you want to register it as a replication source.

If you want to create an empty subscription set, leave the Source-to-Target Mapping page empty. You can add subscription-set members to the subscription set later. You can add members to an existing subscription set by using one of the following notebooks:

- **Subscription Set Properties.** Use this notebook if you have already created the subscription set and want to add one or more subscription-set members to it.

From the contents pane for the **Subscription Sets** folder, right-click a subscription set and select **Properties**.

- **Add Members to Subscription Sets.** Use this notebook to add one member to multiple subscription sets. For example, if you select four subscription sets when you open this notebook, you can add one member to each. Each member must use the same source.

In the contents pane for the **Registered Tables** folder, right-click a source table and select **Add Member**.

You can add a registered source to multiple subscription sets by using the Add Members to Subscription Sets notebook. Thus, you can create several empty subscription sets and populate them all with the same source-target mappings. All of the subscription sets selected for the Add Members to Subscription Sets notebook must use the same Capture server and Capture schema.

## Scheduling the subscription set

After you map sources to targets (or create an empty subscription set), you define subscription-set timing information. On the Schedule page of the Create Subscription Set window, specify when the subscription set should first be eligible for processing; the default is the current date and time of the local machine. You also specify the timing for how often the subscription set should be eligible for processing:

- **Time-based replication**

The Apply program will process this subscription set using a regular time interval.

- **Event-based replication**

The Apply program will process this subscription set whenever an event occurs.

- **Both time-based and event-based replication**

The Apply program will process this subscription set using both a regular time interval and whenever an event occurs. In this case, the subscription set will be eligible for processing at both the scheduled time and when the event occurs.

## Adding SQL statements or stored procedures to the subscription set

After you define subscription-set timing information, you can optionally add SQL statements or stored procedures to the subscription set. On the

Statements page of the Create Subscription Set window, you can add SQL statements or stored procedures that the Apply program should run while processing the subscription set. Click **Add** to add a statement or procedure to the subscription set.

In the Add SQL Statement or Procedure Call window, you can enter an SQL statement or use SQL Assist to define the statement. You can specify that these statements or procedures should run at the target server, before or after processing the subscription set, or at the Capture control server before processing the subscription set. You can also add SQLSTATE values that the Apply program will accept as successful, such as 02000 when attempting to delete a row that does not exist. Because these SQLSTATE values are treated as successful, the error condition does not appear in the Apply trail table (IBMSNAP\_APPLYTRAIL), and the Replication Alert Monitor does not generate an alert for them.

---

## Activating or deactivating subscription sets

Usually, you want your subscription sets to be active so that they can be processed by the Apply program. However, there are times when you might want to deactivate a subscription set for a short time, or indefinitely. If you deactivated a subscription set when you created it, you will probably want to activate it at some time.

To deactivate an active subscription set:

1. Click the **Subscription Sets** folder to display the subscription sets in the contents pane.
2. Right-click an active subscription set, and select **Deactivate**. The Replication Center deactivates the subscription set immediately.

To activate an inactive subscription set:

1. Click the **Subscription Sets** folder to display the subscription sets in the contents pane.
2. Right-click an active subscription set, and select one of the following options:
  - **Activate → Indefinitely** to activate the subscription set.
  - **Activate → One-time only** to activate the subscription set for only one Apply cycle.

The Replication Center activates the subscription set immediately.

---

## Promoting replication objects

After you register sources and create subscription sets on a database server, you might want to copy the replication definitions to another database (for example, from a test system to a production system) without having to re-register the sources or re-create the subscription sets. The Replication Center provides Promote functions to help you copy replication definitions from one database to another.

### Restrictions:

- You can use the Promote functions to copy replication definitions only between *like* systems, for example from one DB2 for UNIX and Windows system to another DB2 for UNIX and Windows system, but not from a DB2 for UNIX and Windows system to a DB2 for z/OS system. As long as the systems are all the same type of operating-system platform, you can use the Promote functions for OS/400, UNIX, Windows, or z/OS systems.
- You cannot use the Promote functions to copy replication definitions for non-DB2 databases or federated database objects.
- You cannot use the Promote functions to copy replication definitions that include OS/400 remote journals.

## Promoting registered tables or views

To promote registered tables:

1. Click the **Registered Tables** folder to display the registered source tables in the contents pane.
2. Right-click a source table and select **Promote**. The Promote Registered Tables window opens.
3. In the Promote Registered Tables window, specify the information for the database server to which you want to copy the registration information:
  - Capture control server alias  
Select the new Capture control server for the registered source table.
  - Capture schema  
Specify the new Capture schema for the registered source table.
  - CD table schema  
Specify the new schema name for the CD table that is associated with the source table.
  - Table schema  
Specify the new schema name for the table. You can use the Promote function to create the source table in the new database.

To promote registered views, click the **Registered Views** folder to display the registered source views in the contents pane, right-click a source view and select **Promote**.

## Promoting subscription sets

To promote subscription sets:

1. Click the **Subscription Sets** folder to display the subscription sets in the contents pane.
  2. Right-click a subscription set and select **Promote**. The Promote Subscription Set window opens.
  3. In the Promote Subscription Set window, specify the information for the database server to which you want to copy the subscription-set information:
    - **Apply control server alias**  
Select the new Apply control server for the subscription set. You can select the Apply control server that is already defined for the subscription set.
    - **Capture control server alias**  
Select the new Capture server for the subscription set. You can select the Capture control server that is already defined for the subscription set.
    - **Target server alias**  
Select the new target server for the subscription set. You can select the target server that is already defined for the subscription set.
    - **Apply qualifier**  
Type a new Apply qualifier for the subscription set.
    - **Subscription set name**  
Type a new name for the subscription set.
    - **Capture schema**  
Type a new Capture schema for the source tables in the subscription set.
    - **Schema name for the source table or view**  
Type a new schema name for the source tables in the subscription set.
    - **Schema name for the target table or view**  
Type a new schema name for the target tables in the subscription set.
- You can leave any field blank if you want to use the value from the current subscription-set definition.

---

## Forcing a full refresh of target tables

There are times when you might need to reload a target table. For example, a gap in the source database log or journal might cause the Capture program to stop and request a cold start, which requires a full refresh of all target tables based on that source database. For small tables, you can let the Apply program perform the full refresh automatically. For large tables, you should use the ASNLOAD exit routine.

Using the Replication Center, you can bypass the full refresh that is normally done by the Apply program so that you can perform an unload or extract from the source table and a load to the target table. The Replication Center makes the necessary changes to the replication control tables to ensure that replication continues to run after the load is complete.

To perform a manual full refresh:

1. Click the **Subscription Sets** folder to display the subscription sets in the contents pane.
2. Right-click a subscription set and select **Full Refresh → Manual**.
3. Read the text in the Full Refresh – Manual Introduction window and click **Next**.
4. Click **Next** in the subsequent windows until you have completed the task.

The Full Refresh – Manual window helps you perform the following steps:

1. Disable current subscriptions for the selected subscription sets.  
After the subscription sets are disabled, you can unload the source table and load the target table.
2. Re-enable the subscriptions for the selected subscription sets.

You can run the generated SQL scripts for steps 1 and 2 immediately, or at a later time. Be sure to perform these steps in the order suggested by the Full Refresh – Manual window, or your replication environment might produce unpredictable results.

To perform an automatic full refresh, so that the Apply program will initiate the full refresh during the next Apply cycle:

1. Click the **Subscription Sets** folder to display the subscription sets in the contents pane.
2. Right-click a subscription set and select **Full Refresh → Automatic**.

---

## Removing or deleting replication definitions

You can use the Replication Center to remove or delete any of the replication definitions that you create. You can perform any of the following tasks:

- Remove user IDs from the Replication Center
- Drop replication control tables from a Capture control server, an Apply control server, or a Monitor control server (not applicable for an OS/400 system)
- Remove a Capture control server, an Apply control server, or a Monitor control server from the Replication Center
- Delete registrations for source tables or views
- Delete subscription sets

- Delete members from subscription sets
- Delete statements from subscription sets
- Remove stored procedures from subscription sets

See the Replication Center online help for information about each of these tasks.

---

## Operating the Capture program

You can perform many daily operations tasks for replication from the Replication Center. For example, you can start or stop the Capture program. To operate the Capture program, expand the **Operations** folder, then click the **Capture Control Servers** folder to display the currently defined Capture control servers in the contents pane. Right-click one of the Capture control servers, and select one of the following operational tasks:

- Start the Capture program
- Stop the Capture program
- Suspend the Capture program
- Resume the Capture program (after suspending it)
- Initiate the Capture pruning process to prune the following tables: CD, UOW, Capture monitor, Capture trace, and signal
- Reinitialize the Capture program so that it re-reads the register table
- View or change the values stored in the Capture parameters table
- View or change the current parameters that the Capture program is using
- View messages issued by the Capture program
- View statistics gathered by the Capture program:
  - The number of rows that the Capture program inserted into the CD table or skipped
  - The number of rows that the Capture program pruned from CD tables
  - The number of transactions that the Capture program committed
  - How much memory the Capture program is using
- View the average latency of the Capture program
- Query the status of the Capture program

You can perform any of these tasks for a Capture program that is running anywhere in your replication network.

---

## Operating the Apply program

You can also use the Replication Center to operate the Apply program. To operate the Apply program, expand the **Operations** folder, click the **Apply Control Servers** folder, expand one of the Apply control servers, and click the **Apply Qualifiers** folder to display currently defined Apply qualifiers in the contents pane. Right-click one of the Apply qualifiers, and select one of the following operational tasks:

- Start the Apply program
- Stop the Apply program
- Display a report for subscription-set activity:
  - Display all subscription sets
  - Display failed subscription sets
  - Display successful subscription sets
  - Display an error summary report for each failed subscription set
- Display performance information for the Apply program:
  - Display the number of rows fetched from CD tables
  - Display the elapsed time for each subscription set
- Display a report on the end-to-end latency for each subscription set
- Query the status of the Apply program

You can perform any of these tasks for an Apply program that is running anywhere in your replication network.

---

## Operating the Replication Alert Monitor

You can use the Replication Center to define contacts and alert conditions for the Replication Alert Monitor.

To create contacts that will be notified when the Replication Alert Monitor detects any of the specified alert conditions:

1. Expand the **Operations** folder.
2. Expand the **Monitor Control Servers** folder.
3. Expand a Monitor control server.
4. Right-click the **Contacts** folder, and select **Create Contact → Person** or **Create Contact → Group**.
5. In the Create Contact window, specify the person's name and e-mail or pager address. In the Create Contact Group window, specify the name of the group and the members of the group.

To select alert conditions for the Capture program:

1. Expand the **Operations** folder.



2. Expand the **Monitor Control Servers** folder.
3. Expand a Monitor control server.
4. Right-click the **Monitor Qualifiers** folder, and select **Select Alert Conditions for Capture Schemas**.
5. In the Select Alert Conditions for Capture Schemas window, specify the following information:
  - The Monitor qualifier that is associated with the instance of the Replication Alert Monitor
  - The Capture control server that you want to monitor
  - The Capture schema that you want to monitor
  - Any alert conditions

To select alert conditions for the Apply program:

1. Expand the **Operations** folder.
2. Expand the **Monitor Control Servers** folder.
3. Expand a Monitor control server.
4. Right-click the **Monitor Qualifiers** folder, and select **Select Alert Conditions for Apply Qualifiers or Subscription Sets**.
5. In the Select Alert Conditions for Apply Qualifiers or Subscription Sets window, specify the following information:
  - The Monitor qualifier that is associated with the instance of the Replication Alert Monitor
  - The Apply control server that you want to monitor
  - Any specific subscription sets that you want to monitor
  - Any specific Apply qualifiers that you want to monitor
  - Any alert conditions

To start the Replication Alert Monitor for a monitor qualifier:

1. Expand the **Operations** folder.
2. Expand the **Monitor Control Servers** folder.
3. Expand a Monitor control server.
4. Expand the **Monitor Qualifiers** folder.
5. Right-click a monitor qualifier and select **Start Monitor**.

You can perform any of these tasks for a Replication Alert Monitor that is running anywhere in your replication network.



---

## Chapter 15. Basic data replication scenario: DB2 for Windows

Use the scenario in this chapter to gain some experience using the DB2<sup>®</sup> Replication Center and the Capture and Apply programs. Follow the steps in this simple scenario to copy changes from a DB2 replication source to a target table in a database on DB2 for Windows<sup>®</sup> Enterprise Server Edition (ESE) or Workgroup Server Edition (WSE).

The scenario consists of the following parts:

1. “Before you begin”
2. “Planning this scenario” on page 270
3. “Setting up the replication environment for this scenario” on page 272
4. “Operating in a replication environment” on page 286

---

### Before you begin

If you want to work through this scenario on your computer, set up your system using these steps:

1. Make sure that you have DB2 for Windows installed on your computer.
2. Make sure you have created the default DB2 instance. This scenario assumes that all databases are within the same instance.
3. Make sure you have access to the SAMPLE database. This database will be both the source server and the Capture control server for this scenario.

To create the SAMPLE database, use the **First Steps** application: select **Start → Programs → IBM DB2 → Set-up Tools → First Steps**). After the database is created, close the First Steps window.

If you did not install First Steps when you installed DB2, open a DB2 command window and issue the **db2sampl** command to create the SAMPLE database.

4. Use the DB2 Control Center to create a new database called COPYDB, which you will use as the target server and the Apply control server. To create the database, right-click the **Databases** folder, select **Create → Database Using Wizard**, and follow the instructions for creating a new database with default options. Both the name and the alias for the database should be COPYDB.

The steps in this chapter use the data in the DEPARTMENT table from the SAMPLE database. The fully qualified name is *schema*.DEPARTMENT; where *schema* is the user ID that created the table. Table 12 on page 270 shows the

DEPARTMENT table.

Table 12. The DEPARTMENT table

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT	LOCATION
A00	SPIFFY COMPUTER SERVICE DIV.	000010	A00	-
B01	PLANNING	000020	A00	-
C01	INFORMATION CENTER	000030	A00	-
D01	DEVELOPMENT CENTER	-	A00	-
D11	MANUFACTURING SYSTEMS	000060	D01	-
D21	ADMINISTRATION SYSTEMS	000070	D01	-
E01	SUPPORT SERVICES	000050	A00	-
E11	OPERATIONS	000090	E01	-
E21	SOFTWARE SUPPORT	000100	E01	-

For the remainder of this scenario, use the user ID with which you created the SAMPLE and COPYDB databases. Because you created the databases, you have the required authority (DBADM or SYSADM) to perform replication tasks.

---

## Planning this scenario

Assume that your group uses an application that generates reports. This application needs information that exists in the DEPARTMENT table of the SAMPLE database. Instead of using the data directly from the source table, you want to copy the changes to a target table that can be read only by the report-generating application. For ease of administration, you want to keep the target table on the same machine as the source table.

You require a simple data distribution configuration, with changes from one replication source being replicated to a single read-only copy. This section describes the design and planning issues that you need to consider before you perform any replication tasks.

### Replication source

You already know that the replication source is the *schema*.DEPARTMENT table in the SAMPLE database. Before you set up your environment, you must decide what you want to replicate from that table; you decide to register all columns and subscribe to all columns.

## Replication target

You decide that you want your replication target to be the COPYDB database, which you created in “Before you begin” on page 269. Currently there is no target table in that database; you want the Replication Center to create the target table according to your specifications. This method of automatically generating a target table is preferred because it ensures correct mapping to the replication source. You can instead use existing target tables, but this scenario assumes that the target table does not exist.

Assume that you want the target table in COPYDB to contain the columns of information shown in Table 13.

*Table 13. Columns for the COPYDB table*

DEPTNO	Information from the DEPTNO column in the replication source table. This column will be the primary key of the target table.
DEPTNAME	Information from the DEPTNAME column in the replication source table.
MGRNO	Information from the MGRNO column in the replication source table.
ADMRDEPT	Information from the ADMRDEPT column in the replication source table.
LOCATION	Information from the LOCATION column in the replication source table.

Because the columns in the target table simply reflect the data from the source table, and because there will be only one row in the target table for each row in the source table, you can use a *user copy* type of target table.

## Replication options

For the purpose of this scenario, you decide to store the CD table, the target table, and the replication control tables in their respective default table spaces, as shown in Table 14. Although the SAMPLE and COPYDB databases exist in the same machine, their table spaces are in separate containers.

*Table 14. Tables and table spaces used in this scenario*

Database	Tables	Table spaces	Contents
SAMPLE	<i>schema</i> .DEPARTMENT	USERSPACE1	Source table
	<i>schema</i> .CDDEPARTMENT	TSCDDEPARTMENT	The CD table for the DEPARTMENT table
	Capture control tables	TSASNCA and TSASNUOW	The replication control tables for the Capture program

Table 14. Tables and table spaces used in this scenario (continued)

Database	Tables	Table spaces	Contents
COPYDB	<i>schema</i> .TGDEPTCOPY	TSTGDEPTCOPY	Target table
	Apply control tables	TSASNAA	The replication control tables for the Apply program
	Monitor control tables	REPLMONT1, REPLMONT2, and REPLMONT3	The replication control tables for the Replication Alert Monitor

Typically, you should create the CD table in a separate table space from the source table to reduce potential contention at the table-space level. You should accept the defaults (or define a profile within the Replication Center) for the table spaces of the replication control tables. For a production environment, it is best if you create each table space on a separate device, to reduce potential contention.

For scheduling replication, assume that you want DB2 replication to check for any changes from the source table every minute and replicate them to the target table. Although a report-generating application doesn't require that kind of response time, you want to test the replication environment to make sure that everything is working correctly.

Also, you decide that after each replication cycle, you want to delete any records from the Apply trail table that are older than one week (seven days). This pruning prevents the table from growing too large.

## Setting up the replication environment for this scenario

After planning the replication model, you are ready to set up the replication environment. Because almost all of the following steps use the Replication Center, be sure that it is running: from the Windows **Start** menu: select **Programs → IBM DB2 → General Administration Tools → Replication Center**.

### Step 1: Create replication control tables for the Capture program

The Capture program reads the replication control tables for current registration information and stores its status in these tables. Any database that will act as a Capture control server must contain the Capture control tables.

#### *To create Capture control tables:*

1. Expand the **Replication Definitions** folder.
2. Right-click the **Capture Control Servers** folder and select **Create Capture Control Tables → Quick**. Alternatively, you could customize the Capture control tables by selecting **Create Capture Control Tables → Custom**.

3. In the Select a Server window, select the SAMPLE database. This database will be your Capture control server. Click **OK**.
4. In the Create Control Tables - Quick - Server Information window, select **Host sources for replication and capture changes to those sources**. Then click **Next**.
5. In the Create Control Tables - Quick - Replication Details window, click **Next**. You do not need to change any of the information in this window.
6. In the Create Control Tables - Quick - Table Spaces window, enter the table space specification for the TSASNCA table space. For example, set the buffer pool to IBMDEFAULTBP. For this scenario, accept the default Capture schema, ASN.
7. In the Create Control Tables - Quick - Table Spaces window, enter the table space specification for the TSASNUOW table space.
8. After you enter information for both table spaces in the Create Control Tables - Quick - Table Spaces window, click **OK**.
9. Click **Close** on the Message Dialog window. This window shows the result of generating the SQL script that will create the Capture control tables. If there were any errors, they would be displayed in this window.
10. Enter a valid user ID and password in the Run Now or Save SQL window and click **OK** to run the SQL script immediately.
11. You should see a message in the DB2 Message window that the script ran successfully. Click **Close**.
12. Expand the **Capture Control Servers** folder. The SAMPLE database should be displayed under the folder.

## Step 2: Enable the source database for replication

The Capture program reads the DB2 log for log records that include changes to registered tables. The log must be an archive log so that the log file will not be reused by DB2 before the Capture program can read the log. For UNIX<sup>®</sup> and Windows environments, the DB2 default is circular logging, so you must change this setting to archive logging.

### *To enable the source database for replication:*

1. Expand the **Capture Control Servers** folder.
2. Right-click the SAMPLE database and select **Enable Database for Replication**.
3. Click **OK** on the Enable Database for Replication window to use archive logging for the SAMPLE database and to initiate a backup for the database. You can optionally use the Database Backup wizard to guide you through the backup process.
4. In the Backup Database window, specify the information for the database backup, and click **Backup Now**.

After you backup the database, you could start the Capture program, but do not start it yet. If you want to start the Capture program, see “Step 7: Replicate the scenario data” on page 284.

### **Step 3: Register a replication source**

After you create the Capture control tables and enable the database for replication, register the DEPARTMENT table as a replication source.

#### ***To register a table as a replication source:***

1. Expand the **Replication Definitions** folder.
2. Expand the **Capture Control Servers** folder.
3. Expand the SAMPLE database folder.
4. Right-click the **Registered Tables** folder and select **Register Tables**.
5. In the Add Registerable Tables window, click **Retrieve All** to list all the tables in the SAMPLE database that you can register as replication sources. Select the DEPARTMENT table and click **OK**. The Register Tables window is displayed, as shown in Figure 8 on page 275.



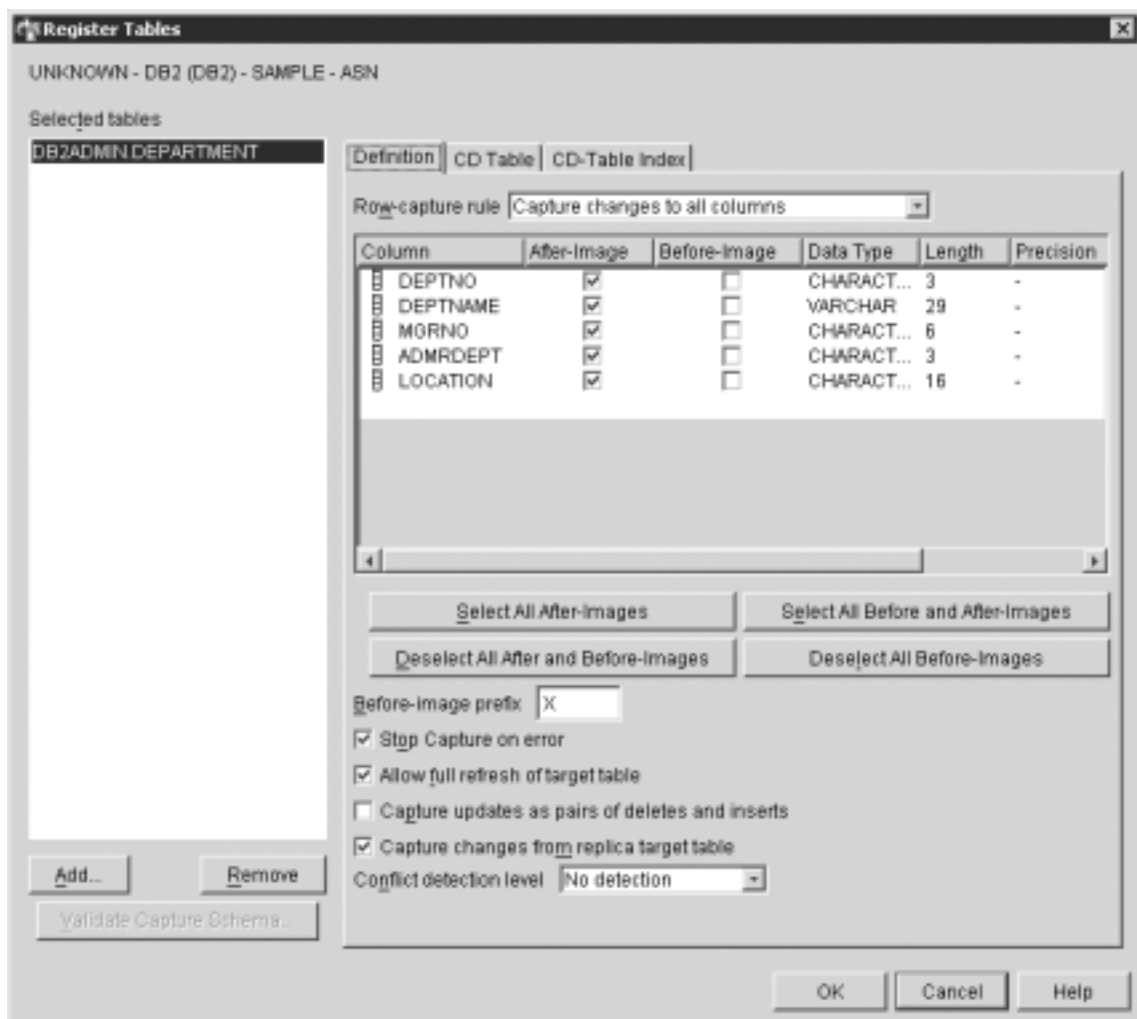


Figure 8. The Register Tables window

6. In the Register Tables window, click the **CD Table** notebook tab. Specify the following information for the CD table space:
  - In the **Specification for table space** area, click the **Container name** field to specify the container name for the TSCDDEPARTMENT table space.
  - In the **Specification for table space** area, change the **Size** field to 1.
  - In the **Specification for table space** area, change the **Unit** field to MB.
  - Specify the other information for this new table space; for example, set the buffer pool to IBMDEFAULTBP.

After you have entered the table-space information, click **OK**.

7. Click **Close** on the Message Dialog window. This window shows the result of generating the SQL script that will create the Capture control tables. If there were any errors, they would be displayed in this window.
8. Click **OK** on the Run Now or Save SQL window to run the SQL script immediately.
9. You should see a message in the DB2 Message window that the script ran successfully. Click **Close**.
10. The contents pane for the SAMPLE database folder should now show the DEPARTMENT table as a registered table. See Figure 9 for an example of the contents pane for the SAMPLE database folder with the DEPARTMENT table as a registered table.

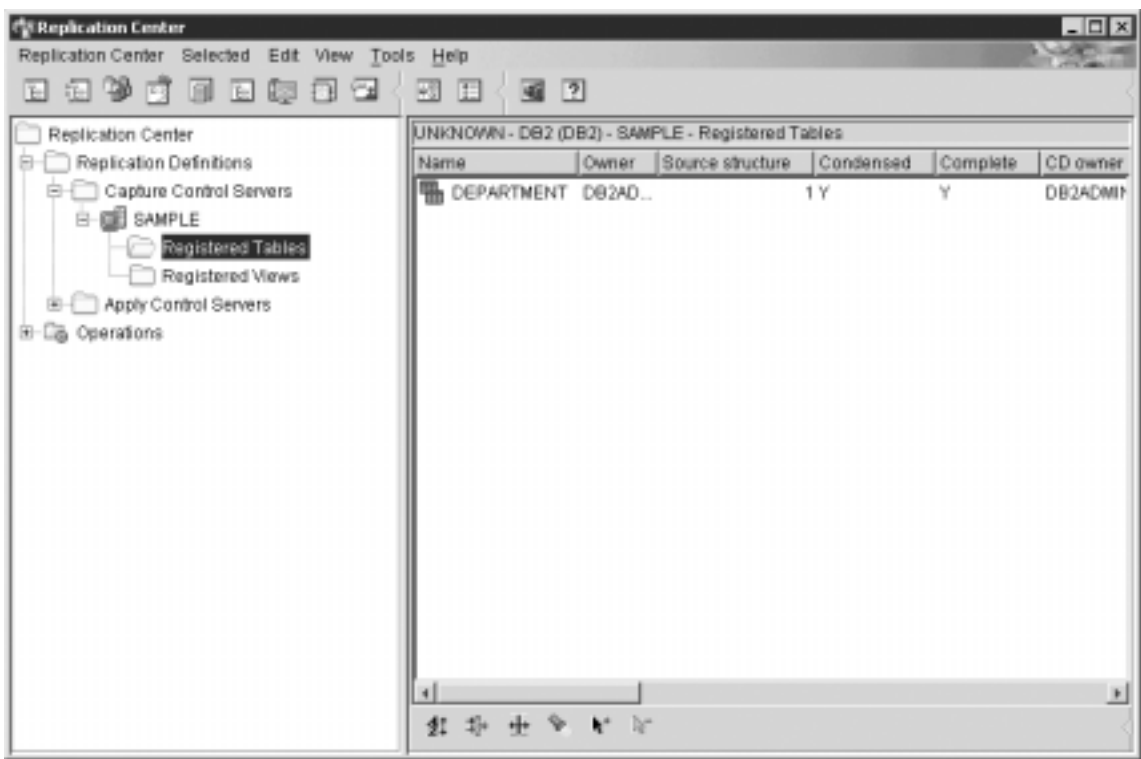


Figure 9. The DEPARTMENT table is listed as a registered table for the SAMPLE database

The DEPARTMENT table is now defined as a replication source. When you ran the SQL script, the Replication Center created the CD table and CD-table index for this replication source, and it updated the Capture control tables.

#### Step 4: Create replication control tables for the Apply program

The Apply program reads the replication control tables for current subscription-set information and stores its status in these tables. Any database that will act as an Apply control server must contain the Apply control tables.

##### *To create Apply control tables:*

1. Expand the **Replication Definitions** folder.
2. Right-click the **Apply Control Servers** folder and select **Create Apply Control Tables → Quick**. Alternatively, you could customize the Apply control tables by selecting **Create Apply Control Tables → Custom**.
3. In the Select a Server window, select the COPYDB database. This database will be your Apply control server. Click **OK**.
4. In the Create Control Tables - Quick - Server Information window, select **Apply captured changes to target tables**. Then click **Next**.
5. In the Create Control Tables - Quick - Replication Details window, click **Next**. You do not need to change any of the information in this window.
6. In the Create Control Tables - Quick - Table Spaces window, enter the table space specification for the TSASNAA table space. For example, set the buffer pool to IBMDEFAULTBP. Click **OK**.
7. Click **Close** on the Message Dialog window. This window shows the result of generating the SQL script that will create the Apply control tables. If there were any errors, they would be displayed in this window.
8. Enter a valid user ID and password in the Run Now or Save SQL window and click **OK** to run the SQL script immediately.
9. You should see a message in the DB2 Message window that the script ran successfully. Click **Close**.
10. Expand the **Apply Control Servers** folder. The COPYDB database should be displayed under the folder.

#### Step 5: Create a subscription set and a subscription-set member

After you register the source table, you need to create a subscription set. A subscription set defines a relationship between the replication source database (SAMPLE in this scenario) and a target database (COPYDB in this scenario). A subscription-set member defines a relationship between the replication source table (DEPARTMENT in this scenario) and one or more target tables (this scenario has only one, it will be called DEPTCOPY).

##### *To create a subscription set and a subscription-set member:*

1. Expand the **Replication Definitions** folder.
2. Expand the **Apply Control Servers** folder.
3. Expand the COPYDB folder.
4. Right-click the **Subscription Sets** folder and select **Create**.

You can also create a subscription set by selecting the **Registered Tables** folder of the SAMPLE database, right-clicking the DEPARTMENT table in the contents pane, and selecting **Create Subscription Set**.

5. In the Set Information page of the Create Subscription Set window, enter the following information:
  - a. In the **Set name** field, enter DEPTSUB. This string identifies the subscription set and must be unique for a particular Apply qualifier.
  - b. In the **Apply qualifier** field, enter DEPTQUAL. This string identifies the replication definitions that are unique to the instance of the Apply program that will run this subscription set.

**Tip:** The Apply qualifier is case-sensitive. If you want the Apply qualifier to be in lowercase characters, you must delimit it when you type it; for example, "deptqual". If you simply type deptqual, the Replication Center converts the value to uppercase characters by default.

- c. Click the browse button for the **Capture control server alias** field. In the Select a Capture Control Server window, select the SAMPLE database and click **OK**.
  - d. Click the browse button for the **Target server alias** field. In the Select a Target Server window, select the COPYDB database and click **OK**. The COPYDB database is both the target server and the Apply control server.
  - e. Select the **Activate the subscription set** check box.

You do not need to change the settings of the other fields in the Set Information page. The Create Subscription Set window should look similar to the window shown in Figure 10 on page 279.

UNKNOWN - DB2 (DB2) - COPYDB

Set Information | Source-to-Target Mapping | Schedule | Statements

Apply control server alias: COPYDB (COPYDB)

Set name: DEPTSUB

Apply qualifier: DEPTQUAL

Capture control server alias: SAMPLE (SAMPLE)

Capture schema: ASN

Target server alias: COPYDB (COPYDB)

☒ Activate the subscription set

☒ Make active indefinitely

☐ Make active for one Apply cycle only

Set processing properties

Data blocking factor: 20

☐ Allow Apply to use transactional processing for set members

Number of transactions applied to target table before Apply commits: 0

OK Cancel Help

Figure 10. The Create Subscription Set window

6. In the Source-to-Target Mapping page of the Create Subscription Set window, enter the following information:
  - a. Click **Add** to add a registered source to the subscription-set member.
  - b. In the Add Registered Sources window, click Retrieve All to display all registered sources in the SAMPLE database.
  - c. In the Add Registered Sources window, select the DEPARTMENT table and click **OK**.

- d. In the Source-to-Target Mapping page of the Create Subscription Set window, change the name of the target table from TGDEPARTMENT to TGDEPTCOPY: select TGDEPARTMENT in the **Target name** column of the Subscription-set members table, and type TGDEPTCOPY over the default name.

Do not change the target type because you want to create a user copy target table.

- e. Click **Details** to open the Member Properties window. From this window, you can define the properties for the subscription-set member.

You do not need to make any changes to the Column Selection or Column Mapping pages of the Member Properties window because you want to replicate all columns and create the same columns in the target table as in the source table. An example of the Member Properties window is shown in Figure 11.

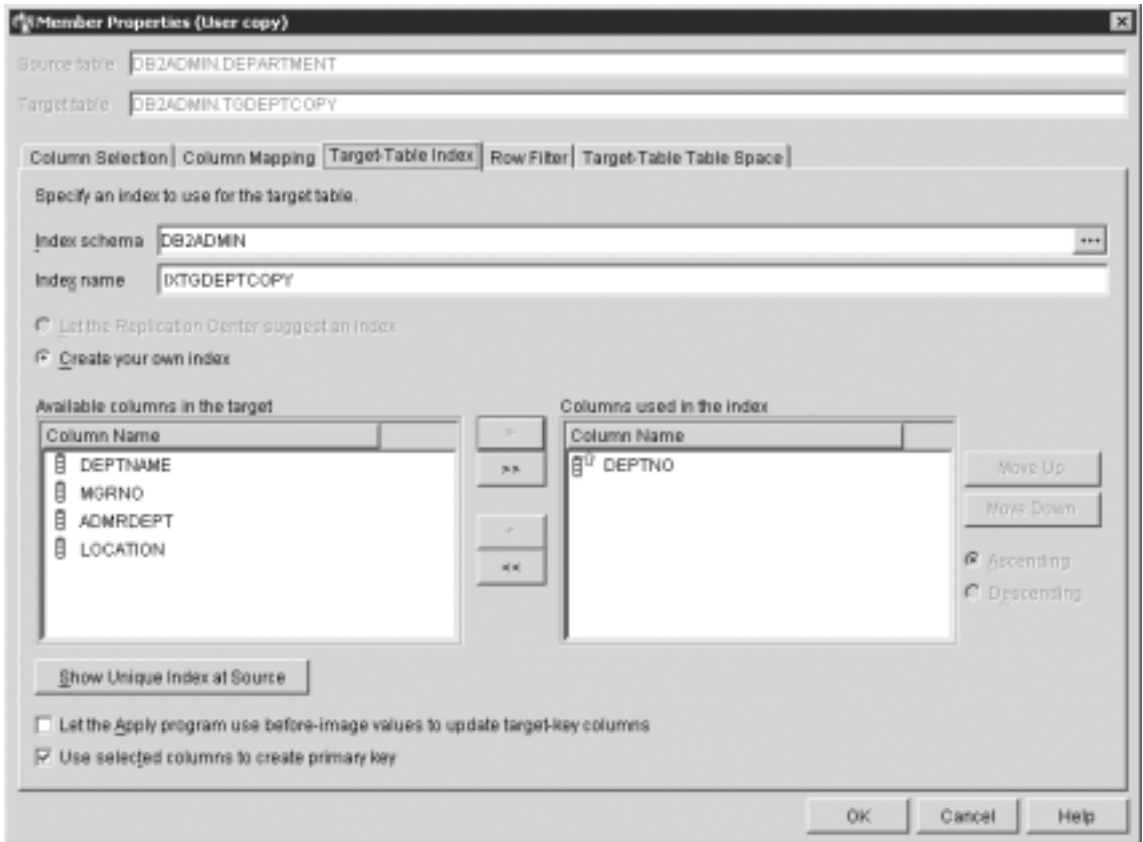


Figure 11. The Member Properties window

7. In the Target-Table Index page of the Member Properties window:
  - a. Select the DEPTNO column from the **Available columns in the target** list.
  - b. Click the move button (>) to move the DEPTNO column to the **Columns used in the index** list.
  - c. Select **Use selected columns to create primary key** to use the DEPTNO column as the primary key for the target table.
8. In the Row Filter page of the Member Properties window, enter the following clause in the **WHERE statement** field:
 

```
DEPTNO >= 'E00'
```

This WHERE clause indicates that you want to replicate only those rows that meet certain criteria; in this case, that the department number is greater than or equal to “E00”. This WHERE clause will cause the target table to contain three rows, instead of all nine rows.

9. In the Target-Table Table Space page of the Member Properties window, specify the following information for the new TSTGDEPTCOPY table space:
  - In the **Specification for table space** area, click the **Container name** field to specify the container name for the TSTGDEPTCOPY table space.
  - In the **Specification for table space** area, change the **Size** field to 1.
  - In the **Specification for table space** area, change the **Unit** field to MB.
  - Specify the other information for this new table space; for example, set the buffer pool to IBMDEFAULTBP.

You can also specify the other information for this new table space; for example, set the buffer pool to IBMDEFAULTBP.

10. Click **OK** to close the Member Properties window.
11. In the Schedule page of the Create Subscription Set window, change the number of minutes to 1 so that the Apply program will process this subscription set every minute. Use the spin button on the **Minutes** field in the **Frequency of replication** area to select 1-minute intervals (or type 1 in the field).
 

Keep the default values for **Start date**, **Start time**, **Time-based**, and **Use relative timing**.
12. In the Statements page of the Create Subscription Set window, click **Add** to open the Add SQL Statement or Procedure Call window. Use this window to define the SQL statements that will be processed when the subscription set is run. In the Add SQL Statement or Procedure Call window, enter the following information:
  - a. In the **SQL Statement** field, enter:

```
DELETE FROM ASN.IBMSNAP_APPLYTRAIL WHERE LASTRUN  
< (CURRENT_TIMESTAMP - 7 DAYS)
```

This statement deletes any records in the Apply trail table that are older than seven days.

The Apply program will execute the SQL statement that you added at the target server after the subscription set is processed. The SQL statement must run at the target server because the Apply control server and target server are co-located and the Apply trail table is in the Apply control server.

**Tip:** The Apply program runs SQL statements or procedures that you add to a subscription set during every subscription cycle. This example is inefficient because the Apply program will execute this statement every minute, even though the statement will only delete data from the APPLYTRAIL table at most once every 24 hours.

- b. In the **SQLSTATE** field, enter 02000 and click **Add**. This SQL state indicates that "row not found" error is acceptable and that the Apply program should ignore these errors.

**Tip:** You can define up to ten SQL states that you want the Apply program to ignore for this subscription set.

- c. Click **OK** to close the Add SQL Statement or Procedure Call window.
13. Click **OK** to close the Create Subscription Set window.
14. Click **Close** on the Message Dialog window. This window shows the result of generating the SQL script that will update the Apply control tables and create the target table. If there were any errors, they would be displayed in this window.
15. Click **OK** on the Run Now or Save SQL window to run the SQL script immediately.

You could save the SQL script to a file for future use and also run it immediately:

- a. Select **Save to file**.
- b. Fill in the information in the **Save specifications** area, such as the file name.
- c. Click **Apply** to save the file. If the script has multiple parts, each part is saved to a separate file using the name you specify plus a number. The Run Now or Save SQL window remains open.
- d. Select **Run now**.
- e. Click **OK** to run the script and close the Run Now or Save SQL window.



You can also save the SQL script to a file to run it later, or you can save the SQL script and run it

16. You should see a message in the DB2 Message window that the script ran successfully at both the SAMPLE and COPYDB servers. Click **Close**.
17. Expand the **Apply Control Servers** folder and the COPYDB folder, then click the **Subscription Sets** folder. The contents pane for the **Subscription Sets** folder should now show the DEPTSUB subscription set. See Figure 12 for an example of the contents pane for the **Subscription Sets** folder with the DEPTSUB subscription set.

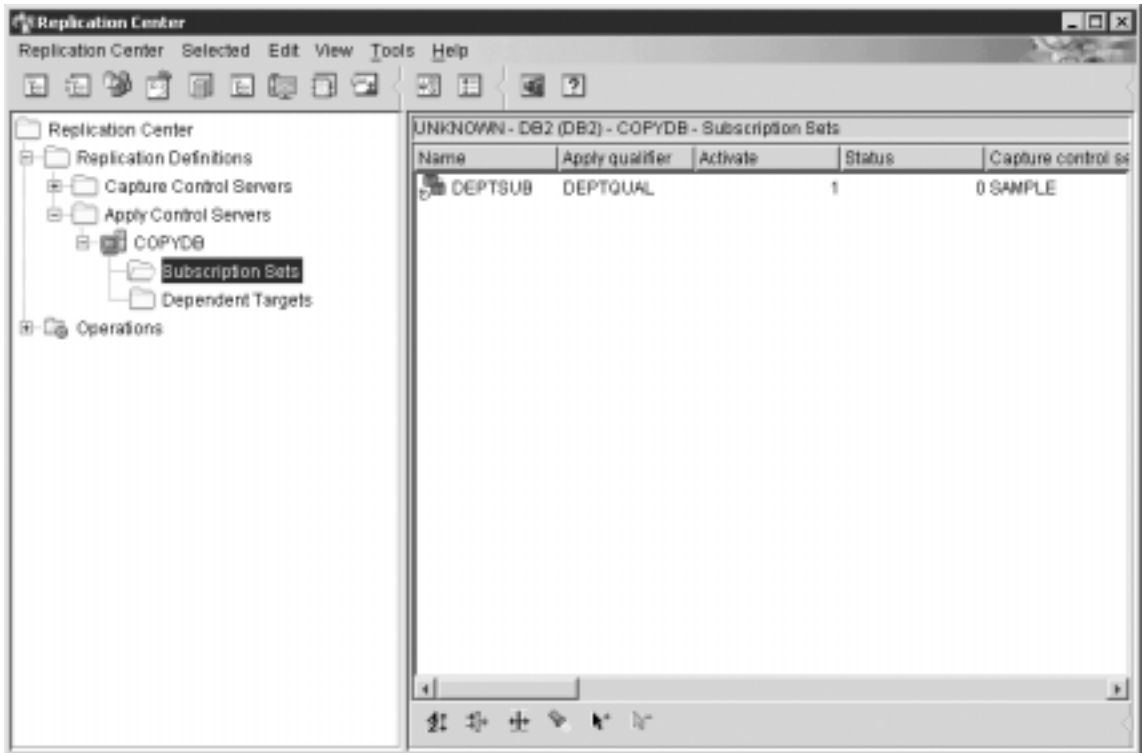


Figure 12. The DEPTSUB subscription set is listed for the COPYDB database

### Step 6: Create an Apply password file

Because the Apply program needs to connect to the Capture control server, the Apply control server, and the target server, you must create a password file for user authentication. Because the contents of the password file are encrypted, only the Apply program can read the file, although you can modify the file.

**To create a password file:**

1. Open a Windows command prompt window and change to the C:\sqllib\bin directory.
2. Enter the following command to create a default password file:  
`asnpwd init using "path"`

where *path* is the fully specified directory path and file name that you want to use when you create the password file. You should see message ASN1981I that confirms that the command completed successfully.

For example, if you want to store the password file in the c:\sqllib\repl directory and name the file asnpwd.aut, enter the following command:

```
asnpwd init using "c:\sqllib\repl\asnpwd.aut"
```

**Tip:** Create the password file in the directory in which you will start the Apply program. When you start the Apply program, you specify the file name for the password file (using the PWDFILE keyword) and the value for the directory in which the Apply program will store its log and work files (using the APPLY\_PATH keyword). One of the Apply program's work files is the password file.

3. Enter the following command to add the user ID and password information for each database to which the Apply program must connect:  
`asnpwd add alias SAMPLE id userid password password using "path"`

where *userid* is a valid DB2 user ID with sufficient authority to update the Capture and Apply control tables. You should see message ASN1981I that confirms that the command completed successfully.

## Step 7: Replicate the scenario data

After you register the replication source and create the subscription set, start the Capture and Apply programs to perform the initial full refresh for the target table and begin change-capture replication.

### ***To start the Capture program:***

1. Expand the **Operations** folder.
2. Select the **Capture Control Servers** folder. The SAMPLE database should be displayed in the contents pane for Capture control servers.
3. Right-click the SAMPLE database and select **Start Capture**.
4. In the Start Capture window, select the CAPTURE\_PATH keyword. In the **Directory in which to store log file** field, enter the directory in which you want the Capture program to write its output, including work and log files. You do not need to change any of the other keywords for the Capture program.
5. Click **OK** on the Start Capture window.

6. Click **OK** on the Run Now or Save Command window to run the command immediately.
7. You should see a message in the DB2 Message window that the command ran successfully. Click **Close**. The Capture program is now running, but will not begin capturing changes for registered tables until the Apply program completes a full refresh for all registered tables.

***To start the Apply program:***

1. Expand the **Operations** folder.
2. Expand the **Apply Control Servers** folder.
3. Expand the COPYDB folder.
4. Select the **Apply Qualifiers** folder. The DEPTQUAL Apply qualifier for subscription set DEPTSUB should be displayed in the contents pane for Apply qualifiers.
5. Right-click the DEPTQUAL Apply qualifier and select **Start Apply**.
6. In the Start Apply window, enter the following information:
  - a. Select a system on which to run the Apply program from the **System** list in the **Where to run Apply** area.
  - b. Select the APPLY\_PATH keyword. Enter a value for the directory in which the Apply program will store its log and work files.
  - c. Select the PWDFILE keyword. Enter a value for the file name of the Apply password file. For example, asnpwd.aut. The path for the password file is the value of the APPLY\_PATH keyword.
  - d. Do not change any of the other keywords for the Apply program.

**Tip:** You can specify the LOADXIT keyword to call the ASNLOAD program. The default ASNLOAD program uses the DB2 EXPORT utility to export data from the source table and uses the DB2 LOAD utility to perform the full refresh for the target table. You can modify ASNLOAD to call any IBM or vendor utility.

7. Click **OK** on the Start Apply window.
8. If necessary, type a valid user ID and password for the system on which you will run the Apply program in the Run Now or Save Command window.
9. Click **OK** on the Run Now or Save Command window to run the command immediately.
10. You should see a message in the DB2 Message window that the command ran successfully. Click **Close**. The Apply program is now running.

If you view the TGDEPTCOPY target table after one replication cycle, you should see results that match the data shown in Table 15. You can use either of the following methods to view the contents of the table:

- Use the DB2 Control Center:
  1. Expand the databases folder for your DB2 instance.
  2. Expand the COPYDB folder.
  3. Select the **Tables** folder.
  4. Right-click the TGDEPTCOPY table in the contents pane and select **Sample Contents**.
- Use the DB2 Command Center or a DB2 command window to issue the following SQL statement:

```
SELECT * FROM schema.TGDEPTCOPY
```

Table 15. The TGDEPTCOPY table

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT	LOCATION
E01	SUPPORT SERVICES	000050	A00	-
E11	OPERATIONS	000090	E01	-
E21	SOFTWARE SUPPORT	000100	E01	-

---

## Operating in a replication environment

After the replication environment is up and running, changes that you make to the replication source table will be replicated to the target table. You can view status for both the Capture and Apply programs to understand your replication latency and other information about your replication environment. And although the Capture and Apply programs can run continuously, there are times when you will want to stop them (for example, to run utilities that use the table spaces that contain the control tables).

### Step 1: Update the source table

Assume that two new departments were created at the Spiffy Computer Service: a technical writing department and a public relations department. Your target table will include both of these departments.

#### *To update the source table:*

1. Select **Start → Programs → IBM DB2 → Command Window** to open a DB2 command window.
2. Connect to the source server:

```
DB2 CONNECT TO SAMPLE
```
3. Add two new rows, one for each department, by typing each of the following commands and pressing Enter after each one:

```
DB2 INSERT INTO DEPARTMENT
VALUES ('F01','TECHNICAL WRITING','000110','F01',NULL)
DB2 INSERT INTO DEPARTMENT
VALUES ('G01','PUBLIC RELATIONS','000120','G01',NULL)
DB2 COMMIT
```

4. Connect to the target server:

```
DB2 CONNECT TO COPYDB
```

5. Wait at least one minute, then verify that the new rows are replicated to the target database by typing the following command and pressing Enter:

```
DB2 SELECT * FROM TGDEPTCOPY
```

You must wait for one minute because the subscription set is eligible for replication every minute. If there were a large amount of data, you might want to wait a bit longer for the Apply to apply that data to the target table.

Table 16 shows the results of the replication, with two new rows appended to the table.

*Table 16. The TGDEPTCOPY table after changes are replicated*

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT	LOCATION
E01	SUPPORT SERVICES	000050	A00	-
E11	OPERATIONS	000090	E01	-
E21	SOFTWARE SUPPORT	000100	E01	-
F01	TECHNICAL WRITING	000110	F01	-
G01	PUBLIC RELATIONS	000120	G01	-

## Step 2: View status for the Capture program

Use the Replication Center to view the following status information for the Capture program:

- Error messages issued by the Capture program
- An analysis of the Capture program's throughput
- A summary of the Capture program's current latency
- The current operational status for the Capture program

Each of these kinds of status give you a snapshot view of how the Capture program is running.

### *To query the status of the Capture program:*

1. Expand the **Operations** folder.
2. Select the **Capture Control Servers** folder.

3. Right-click the SAMPLE database in the contents pane and select **Query Status**.
4. Click **Retrieve** to view current information.

*To view an analysis of the Capture program's throughput:*

1. Expand the **Operations** folder.
2. Select the **Capture Control Servers** folder.
3. Right-click the SAMPLE database in the contents pane and select **Show Capture Throughput Analysis**.
4. In the Capture Throughput Analysis window, you can view the following information:
  - The number of rows inserted to the CD tables from the DB2 log or skipped for various reasons, such as the setting of the CHGONLY keyword
  - The number of rows pruned from the CD tables
  - The number of transactions committed by the Capture program
  - The memory usage of the Capture program within a specific time interval
5. Click **Retrieve** to view current information.

*To view a summary of the Capture program's current latency:*

1. Expand the **Operations** folder.
2. Select the **Capture Control Servers** folder.
3. Right-click the SAMPLE database in the contents pane and select **Show Capture Latency**.
4. In the Capture Latency window, you can view the average, minimum, and maximum latency for the Capture program within a specific time interval.
5. Click **Retrieve** to view current information.

### **Step 3: View status for the Apply program**

Use the Replication Center to view the following status information for the Apply program:

- A summary of subscription-set information, including successful and failed subscription sets
- A summary of the performance of the Apply program
- A summary of the end-to-end replication latency
- The current operational status for the Apply program

Each of these kinds of status give you a snapshot view of how the Apply program is running.

*To query the status of the Apply program:*

1. Expand the **Operations** folder.
2. Select the **Apply Control Servers** folder.
3. Expand the COPYDB folder.
4. Select the **Apply Qualifiers** folder.
5. Right-click the DEPTQUAL Apply qualifier in the contents pane and select **Query Status**.
6. Click **Retrieve** to view current information.

***To view a summary of Apply program performance:***

1. Expand the **Operations** folder.
2. Expand the **Apply Control Servers** folder.
3. Expand the COPYDB folder.
4. Select the **Apply Qualifiers** folder.
5. Right-click the DEPTQUAL Apply qualifier in the contents pane and select **Show Apply Throughput Analysis**.
6. In the Show Apply Throughput Analysis window, you can view the following information:
  - The number of rows fetched by the Apply program from the CD tables
  - The elapsed time for each subscription set
7. Click **Retrieve** to view current information.

***To view a summary of the end-to-end replication latency:***

1. Expand the **Operations** folder.
2. Expand the **Apply Control Servers** folder.
3. Expand the COPYDB folder.
4. Select the **Apply Qualifiers** folder.
5. Right-click the DEPTQUAL Apply qualifier in the contents pane and select **Show End-to-End Latency**.
6. In the Show End-to-End Latency window, you can view the average latency for each subscription set within a specific time interval.
7. Click **Retrieve** to view current information.

#### **Step 4: Stop the Capture and Apply programs**

An important part of maintaining your replication environment is regular database maintenance. Sometimes that maintenance will require you to stop the Capture and Apply programs. For example, you must stop the Capture and Apply programs before you run utilities that directly use the table spaces that are used by these programs.

***To stop the Capture program:***

1. Expand the **Operations** folder.

2. Select the **Capture Control Servers** folder.
3. Right-click the SAMPLE database in the contents pane and select **Stop Capture**.
4. Click **OK** on the Stop Capture window.
5. Click **OK** on the Run Now or Save Command window to run the command immediately.
6. You should see a message in the DB2 Message window that the command ran successfully. Click **Close**. The Capture program is now stopped.

***To stop the Apply program:***

1. Expand the **Operations** folder.
2. Expand the **Apply Control Servers** folder.
3. Expand the COPYDB folder.
4. Select the **Apply Qualifiers** folder.
5. Right-click the DEPTQUAL Apply qualifier in the contents pane and select **Stop Apply**.
6. In the Stop Apply window, click **OK**.
7. In the Run Now or Save Command window, type the directory in which you started the Apply program (the value that you specified for the APPLY\_PATH keyword) in the **Directory** field, or use the browse button to select the path. Click **OK**.
8. Click **OK** on the Run Now or Save Command window to run the command immediately.
9. You should see a message in the DB2 Message window that the command ran successfully. Click **Close**. The Apply program is now stopped.

You can run DB2 utilities on your database now that you have stopped the Capture and Apply programs. Running utilities is beyond the scope of this scenario.

---

## Monitoring replication

After the replication environment is up and running, there will be times when you want to understand how well the Capture and Apply programs are running. You might also want to set up automatic notifications in the event of certain kinds of replication errors.

You can use the Replication Center to query the status of the Capture and Apply programs and to view certain statistics that can tell you how well either program is running. You can also set up the Replication Alert Monitor to notify you when either the Capture or Apply programs encounter certain kinds of replication errors.



## Step 1: Create replication control tables for the Monitor program

The Replication Alert Monitor program reads the replication monitor control tables for current monitor information and stores its status in these tables. Any database that will act as a Monitor server must have the Monitor control tables.

### *To create Monitor control tables:*

1. Expand the **Operations** folder.
2. Right-click the **Monitor Control Servers** folder and select **Create Monitor Control Tables**.
3. In the Select a Server window, select the COPYDB database. This database will be your Monitor control server.
4. In the Create Monitor Control Tables window, select the IBMSNAP\_CONTACTS control table and fill in the information for the RELPMONTS1 table space properties. Click the browse button next to the container name to customize the location for this table space. You can also specify other information for the table space, for example, set the buffer pool to IBMDEFAULTBP. The Create Monitor Control Tables window should look similar to the window shown in Figure 13 on page 292.

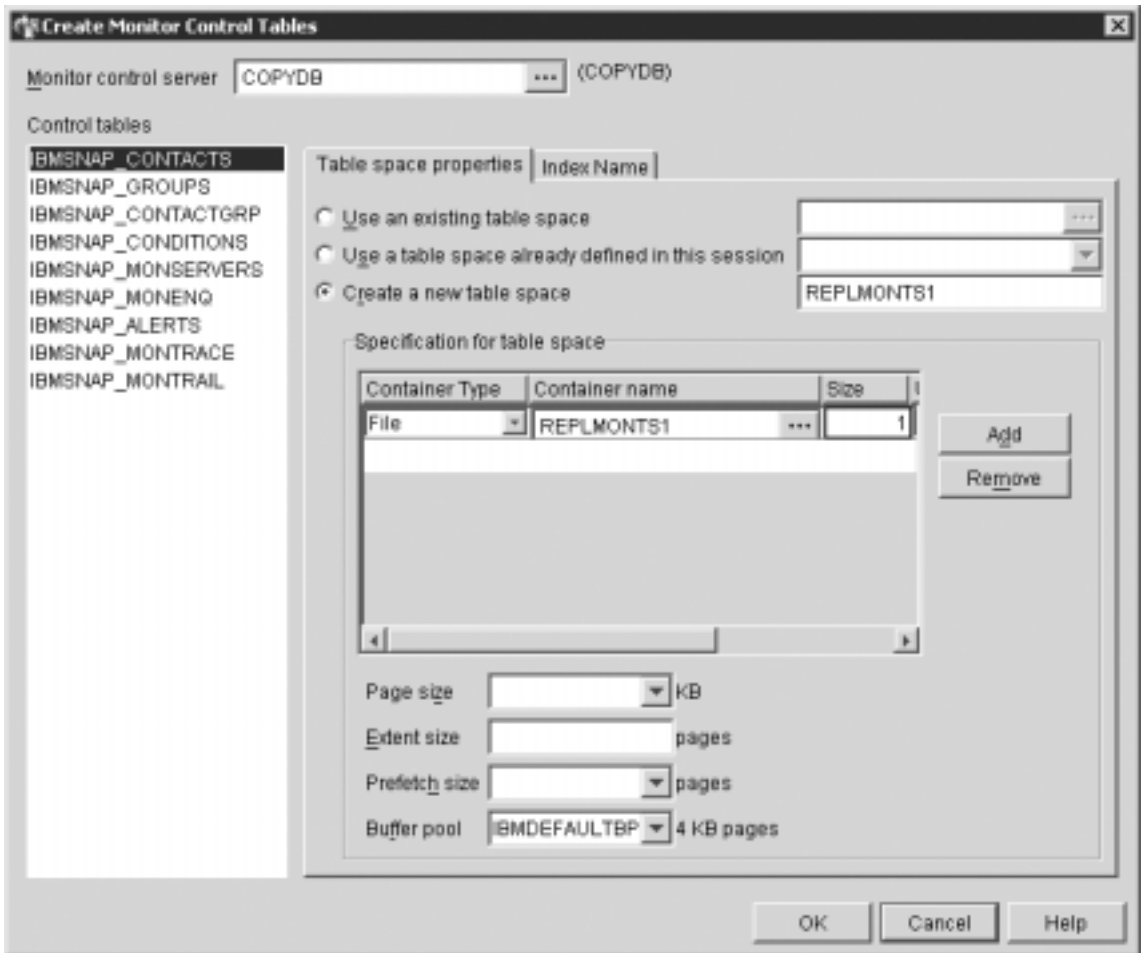


Figure 13. The Create Monitor Control Tables window

5. In the Create Monitor Control Tables window, select the IBMSNAP\_ALERTS control table and fill in the information for the RELPMONTSS2 table space properties.
6. In the Create Monitor Control Tables window, select the IBMSNAP\_MONTRACE control table and fill in the information for the RELPMONTSS3 table space properties.
7. Click **OK** on the Create Monitor Control Tables window to accept the default values for the other control table information, including index names.
8. Click **Close** on the Message Dialog window. This window shows the result of generating the SQL script to create the Monitor control tables. If there were any errors, they would be displayed in this window.

9. Click **OK** on the Run Now or Save SQL window to run the SQL script immediately.
10. You should see a message in the DB2 Message window that the script ran successfully. Click **Close**.
11. Expand the **Monitor Control Servers** folder. The COPYDB database should be displayed under the folder.

### Step 2: Create a contact for replication alerts

The Replication Alert Monitor program can alert you when it detects specific activity of the Capture and Apply programs. You can create individual contacts or groups of contacts if the Replication Alert Monitor should alert several people for a specific alert condition.

#### *To create a contact:*

1. Expand the **Operations** folder.
2. Expand the **Monitor Control Servers** folder.
3. Expand the COPYDB folder.
4. Right-click the Contacts folder and select **Contacts → Person**.
5. In the Create Contact window, enter your name and e-mail address. Click **OK** to close the window.
6. Click **Close** on the Messages and SQL Scripts window. This window shows the result of generating the SQL script to update the Monitor control tables. If there were any errors, they would be displayed in this window.
7. Click **OK** on the Run Now or Save SQL window to run the SQL script immediately.
8. You should see a message in the DB2 Message window that the script ran successfully. Click **Close**.
9. Click the **Contacts** folder. The contact that you defined should be displayed in the contents pane for Contacts.

### Step 3: Select alert conditions for the Capture program

The Replication Alert Monitor program can monitor specific activity of the Capture program. You must select which activities you want to monitor. For each of these activities, you select an alert condition. When the Capture program encounters the condition, the Replication Alert Monitor sends an alert to those contacts that you define for the alert condition.

#### *To create Monitor definitions for the Capture program:*

1. Expand the **Operations** folder.
2. Expand the **Monitor Control Servers** folder.
3. Expand the COPYDB folder.

4. Right-click the **Monitor Qualifiers** folder and select **Select Alert Conditions for Capture Schemas**.
5. In the Select Alert Conditions for Capture Schemas window, enter the following information:
  - a. In the **Monitor qualifier** field, type MON1.
  - b. Click the browse button for the **Capture control server** field to select the Capture control server that you want to monitor. In the Select a Capture Control Server window, select the SAMPLE database and click **OK**.
  - c. Click **Add** to add ASN to the **Selected Capture schemas** list.
  - d. In the Select Capture Schemas window, click **Retrieve All**. Select ASN from the list, and click **OK** to close the window.
  - e. In the **Alert conditions** list, select **Errors**.
  - f. In the Value area, click the browse button for the **Contact** field to select the contact for this alert condition.
  - g. In the Select Contact or Contact Group window, select the contact that you created in step 2 and click **OK** to close the window.
  - h. Click **OK** to close the Select Alert Conditions for Capture Schemas window.
6. Click **Close** on the Messages and SQL Scripts window. This window shows the result of generating the SQL script to update the Monitor control tables. If there were any errors, they would be displayed in this window.
7. Click **OK** on the Run Now or Save SQL window to run the SQL script immediately.
8. You should see a message in the DB2 Message window that the script ran successfully. Click **Close**.
9. Expand the COPYDB folder, expand the **Monitor Qualifiers** folder, and select the MON1 folder. The alert conditions that you defined should be displayed in the contents pane for Monitor qualifiers.

#### **Step 4: Select alert conditions for the Apply program**

The Replication Alert Monitor program can monitor specific activity of the Apply program. You must select which activities you want to monitor. For each of these activities, you select an alert condition. When the Apply program encounters the condition, the Replication Alert Monitor sends an alert to those contacts that you define for the alert condition.

##### ***To create Monitor definitions for the Apply program:***

1. Expand the **Operations** folder.
2. Expand the **Monitor Servers** folder.
3. Expand the COPYDB folder.

4. Right-click the **Monitor Qualifiers** folder and select **Select Alert Conditions for Apply Qualifiers or Subscription Sets**.
5. In the Select Alert Conditions for Apply Qualifiers or Subscription Sets window, enter the following information:
  - a. In the **Monitor qualifier** field, type MON1 if you did not create alert conditions for the Capture program.
  - b. Click the browse button for the **Apply control server** field to select the Apply control server that you want to monitor. In the Select an Apply Control Server window, select the COPYDB database and click **OK**.
  - c. Click **Add** to add DEPTSUB to the select subscription sets list.
  - d. In the Add Subscription Sets window, click **Retrieve All**. Select DEPTSUB from the list and click **OK**.
  - e. In the **Alert conditions** list, select **Full refresh occurred**.
  - f. In the Value area, click the browse button for the **Contact** field to select a contact for this alert condition.
  - g. In the Select Contact or Contact Group window, select the contact that you created in step 2 and click **OK** to close the window.
  - h. Click **OK** in the Select Alert Conditions for Apply Qualifiers or Subscription Sets window.
6. Click **Close** on the Messages and SQL Scripts window. This window shows the result of generating the SQL script to update the Monitor control tables. If there were any errors, they would be displayed in this window.
7. Click **OK** on the Run Now or Save SQL window to run the SQL script immediately.
8. You should see a message in the DB2 Message window that the script ran successfully. Click **Close**.
9. Expand the COPYDB folder, expand the **Monitor Qualifiers** folder, and select the MON1 folder. The alert conditions that you defined should be displayed in the contents pane for Monitor qualifiers. See Figure 14 on page 296 for an example of the contents pane for the **Monitor Qualifiers** folder with the MON1 monitor qualifier.

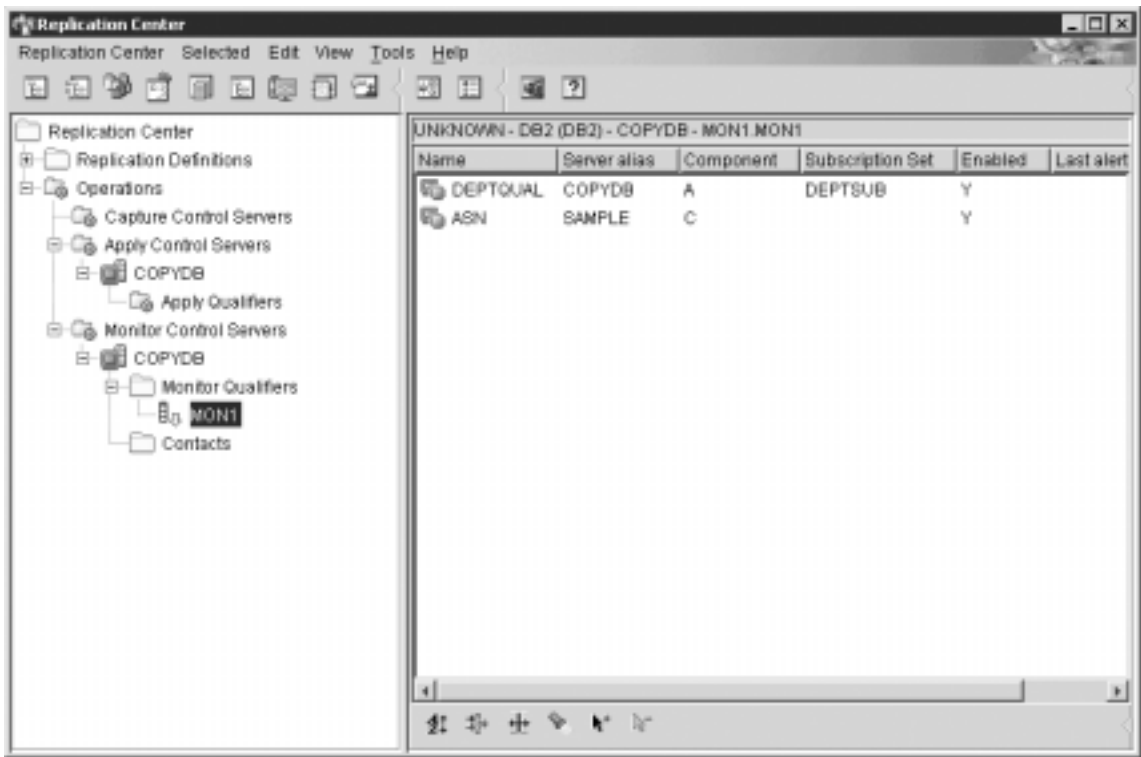


Figure 14. The MON1 monitor qualifier is listed for the COPYDB database

### Step 5: Start the Replication Alert Monitor for a monitor qualifier

After you select alert conditions for the Capture or Apply programs, you can start the Replication Alert Monitor program to monitor the activity of the Capture and Apply programs for the specific conditions associated with a monitor qualifier. When the Capture or Apply program encounters one of the specified conditions, the Replication Alert Monitor sends an alert to those contacts that you defined for the alert condition.

#### ***To start the Replication Alert Monitor:***

1. Expand the **Operations** folder.
2. Expand the **Monitor Servers** folder.
3. Expand the **COPYDB** folder.
4. Expand the **Monitor Qualifiers** folder.
5. Right-click the MON1 monitor qualifier and select **Start Monitor**.
6. In the Start Monitor window, enter the following information:

- a. Select the **MONITOR\_PATH** keyword. Enter a value for the directory in which the Replication Alert Monitor will store its log and work files.

**Tip:** Set the value for the **MONITOR\_PATH** keyword to the value that you set for the **APPLY\_PATH** keyword so that both the Replication Alert Monitor and the Apply program can use the same password file.

- b. Select the **EMAIL\_SERVER** keyword. Enter your e-mail server name.
  - c. Select the **MONITOR\_ERRORS** keyword. Enter your e-mail address so that the Replication Alert Monitor will notify you if it has problems or click the browse button to open the Select Contact or Contact Group window.
  - d. Click **OK** to close the Start Monitor window.
7. Click **OK** on the Run Now or Save Command window to run the command immediately.
  8. You should see a message in the DB2 Message window that the command ran successfully. Click **Close**.

***To display the alerts that the Replication Alert Monitor monitored:***

1. Expand the **Operations** folder.
2. Expand the **Monitor Servers** folder.
3. Expand the **COPYDB** folder.
4. Expand the **Monitor Qualifiers** folder.
5. Select the **MON1** monitor qualifier.
6. In the contents pane for the monitor qualifier, right-click one of the alert conditions and select **Show Alerts**.
7. In the Show Alerts window, specify a time range and click **Retrieve**.

***To stop the Replication Alert Monitor:***

1. Expand the **Operations** folder.
2. Expand the **Monitor Servers** folder.
3. Expand the **COPYDB** folder.
4. Expand the **Monitor Qualifiers** folder.
5. Right-click the **MON1** monitor qualifier and select **Stop Monitor**.
6. Click **OK** on the Run Now or Save Command window to run the command immediately.
7. You should see a message in the DB2 Message window that the command ran successfully. Click **Close**.





---

## Part 3. Replication reference

This part of the book contains the following chapters:

Chapter 16, “Naming rules for replication objects” on page 301 describes how to specify valid names for replication objects.

Chapter 17, “System commands for replication (UNIX, Windows, z/OS)” on page 303 describes commands that experienced DB2 replication users can use instead of the Replication Center for operating replication on UNIX, Windows, and z/OS operating systems.

Chapter 18, “System commands for replication (OS/400)” on page 349 describes the commands that you can use if you want to set up, administer, and maintain replication locally on the OS/400 operating system.

Chapter 19, “Operating the replication programs (z/OS)” on page 453 describes how to start and operate the replication programs using JCL or system-started tasks on z/OS.

Chapter 20, “Using the Windows Service Control Manager to issue system commands (Windows)” on page 459 describes how to start replication programs as services on Windows operating systems.

Chapter 21, “Scheduling replication programs on various operating systems” on page 463 describes how to schedule replication programs on various operating systems.

Chapter 22, “How the DB2 replication components communicate” on page 465 describes how the replication components use the control tables to communicate with each other.

Chapter 23, “Table structures” on page 471 describes the table structures for the replication tables that reside on the various replication servers.

Chapter 24, “Replication messages” on page 549 lists all of the replication messages for UNIX, Windows, and z/OS platforms.



---

## Chapter 16. Naming rules for replication objects

The following table lists the limits for names of replication objects.

*Table 17. Name limits for replication objects*

Object	Name limits
Source and target tables	Follow the naming rules for your database management system.
Source and target columns	Follow the naming rules for your database management system. (Note that all before-image columns have a one-character prefix added to them. To avoid ambiguous before-image column names, ensure that source column names are unique to 29 characters.)
Capture schema	<b>UNIX, Windows, z/OS:</b> The Capture schema can be a string of 30 or fewer characters (except on DB2 for z/OS and OS/390, where it's limited to 18 characters).  <b>OS/400:</b> The Capture schema (CAPCNTLIB) can be a string of 10 or fewer characters.
Apply qualifier	<b>UNIX, Windows, z/OS:</b> The Apply qualifier can be a string of 18 or fewer characters.  <b>OS/400:</b> The Apply qualifier can be a string of 18 or fewer characters but, because Apply jobs can be only up to 10 characters long, the first 10 characters must be unique for a given Apply qualifier.
Monitor qualifier	<b>UNIX, Windows, z/OS:</b> The Monitor qualifier can be a string of 18 or fewer characters.

Also, ensure that you use only these valid characters in names of replication objects:

- A through Z (uppercase letters)
- a through z (lowercase letters)
- numerals (0 through 9)
- the underscore character "\_"

Blanks are not allowed; neither are other special characters such as the colon ":" and the plus sign "+".

Replication system commands and the Replication Center, by default, convert all names that you provide to upper case. Enclose a mixed-case character

name in double quotation marks (or whatever character the target system is configured to use) to preserve the case and save the name exactly as you typed it. For example, if you type `myqual` or `MyQual` or `MYQUAL`, the name is saved as `MYQUAL`. If you type those same names and enclose them in double quotation marks, they are saved as `myqual` or `MyQual` or `MYQUAL`, respectively. Some operating systems don't recognize double quotation marks and you might have to use an escape character, typically a backslash (\).

On Windows operating systems, you *must* use a unique path to differentiate between names that are otherwise identical. For example, assume that you have three Apply qualifiers: `myqual` , `MyQual` , and `MYQUAL`. The three names use the same characters but different case. If these three qualifiers are in the same Apply path, they will cause name conflicts.

**Important:** When setting up Windows services for Capture, Apply, or the Replication Alert Monitor, you must use unique names for the Capture schema, Apply qualifier, and Monitor qualifier. You cannot use case to differentiate names.

---

## Chapter 17. System commands for replication (UNIX, Windows, z/OS)

This chapter describes the replication commands that run under one or more of the following operating systems:

- UNIX
- Windows
- z/OS

All of these commands have a prefix of **asn** and are entered at an operating system command prompt or in a shell script. One of the commands, **asnanalyze**, also works with remote data residing on OS/400 operating systems.

This chapter contains a section for each command. Each section contains a brief description of the command, a syntax diagram, and a table of parameters with corresponding definitions. The end of each section has examples of command usage and cross-references to related information.

The commands include:

- “asnacmd: Operating Apply (UNIX, Windows, z/OS)” on page 304
- “asnanalyze: Operating the Analyzer (UNIX and Windows)” on page 305
- “asnapply: Starting Apply (UNIX, Windows, z/OS)” on page 308
- “asnccap: Starting Capture (UNIX, Windows, z/OS)” on page 316
- “asnccmd: Operating Capture (UNIX, Windows, z/OS)” on page 322
- “asnmcmd: Operating the Replication Alert Monitor (UNIX, Windows, z/OS)” on page 329
- “asnmon: Starting the Replication Alert Monitor (UNIX, Windows, z/OS)” on page 330
- “asnpwd: Maintaining password files (UNIX and Windows)” on page 335
- “asnscrt: Creating a DB2 Replication Service to start Capture, Apply, or the Replication Alert Monitor (Windows only)” on page 338
- “asnscdrop: Dropping a DB2 Replication Service (Windows only)” on page 340
- “asntrc: Operating the replication trace facility (UNIX, Windows, z/OS)” on page 341

asnacmd: Operating Apply (UNIX, Windows, z/OS)

Use the **asnacmd** command to operate the Apply program on UNIX, Windows, and UNIX System Services (USS) on z/OS. Run this command at an operating system prompt or in a shell script.

*To operate the Apply program using the asnacmd command:*

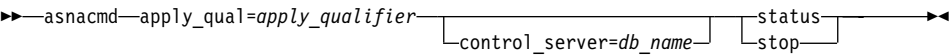


Table 18 defines the invocation parameters.

Table 18. *asnacmd* invocation parameter definitions for UNIX, Windows, and z/OS operating systems

Parameter	Definition
<b>apply_qual=apply_qualifier</b>	<p>Specifies the Apply qualifier that the Apply program uses to identify the subscriptions sets to be served.</p> <p>You must specify an Apply qualifier. The value that you enter must match the value of the APPLY_QUAL column in the subscription set (IBMSNAP_SUBS_SET) table. The Apply qualifier name is case sensitive and can be a maximum of 18 characters.</p>
<b>control_server=db_name</b>	<p>Specifies the name of the Apply control server on which the subscription definitions and Apply control tables reside.</p> <p><b>For UNIX and Windows:</b> If you do not specify an Apply control server, this parameter defaults to the value from the DB2DBDFT environment variable.</p> <p><b>For z/OS:</b> The control server parameter is the name of the database server that connects to the control server.</p>
<b>status</b>	<p>Specify to receive messages that indicate the state of each thread (administration and worker) in Apply.</p>
<b>stop</b>	<p>Specify to stop the Apply program in an orderly way.</p>

Examples for asnacmd

The following examples illustrate how to use the **asnacmd** command.

**Example 1**

To receive messages about the state of each Apply thread:

```
asnacmd apply_qual=AQ1 control_server=dbx status
```

**Example 2**

To stop the Apply program:

```
asnacmd apply_qual=AQ1 control_server=dbx stop
```

**Related tasks:**

- Chapter 19, “Operating the replication programs (z/OS)” on page 453

**Related reference:**

- “ENDDPRAPY: Stopping Apply (OS/400)” on page 397
- “STRDPRAPY: Starting Apply (OS/400)” on page 427

---

**asnanalyze: Operating the Analyzer (UNIX and Windows)**

Use the **asnanalyze** command to generate reports about the state of the replication control tables. This command analyzes replication control tables that reside on any operating system, including OS/400 operating systems; however, you must invoke the command from UNIX or Windows.

You must type a space between the **asnanalyze** command and the first parameter to invoke the command. If you issue the command without any parameters, you receive command help on the screen.

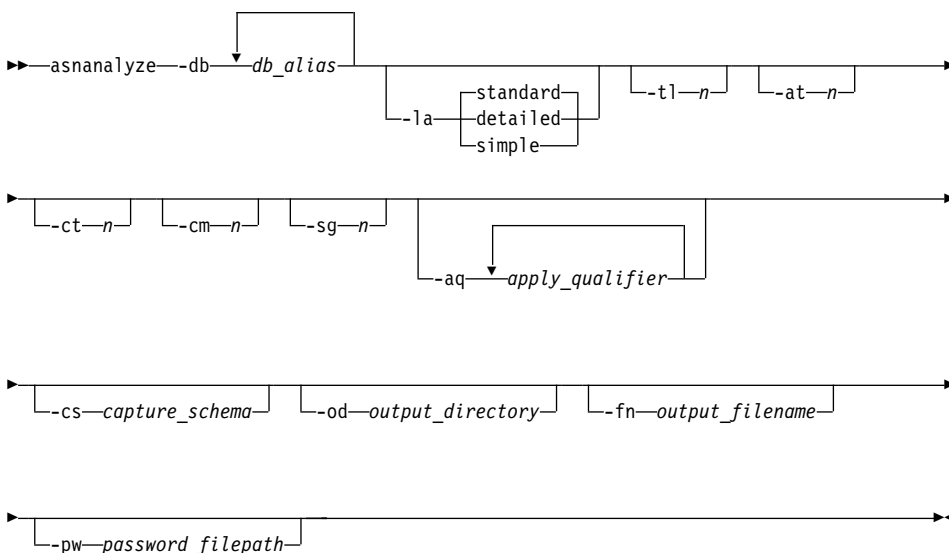
***To operate the Analyzer using the asnanalyze command:***

Table 19 defines the invocation parameters.

*Table 19. asnanalyze invocation parameter definitions for UNIX and Windows operating systems*

Parameter	Definition
<b>-db</b> <i>db_alias</i>	Specifies the Capture control server, target server, and Apply control server.  You must provide at least one database alias. If there is more than one database alias, use blank spaces to separate the values.
<b>-la</b> <i>level_of_analysis</i>	Specifies the level of analysis to be reported:  <b>standard</b> (default) Generates a report that includes the contents of the control tables and status information from the Capture and Apply programs.  <b>detailed</b> Generates the information in the standard report and: <ul style="list-style-type: none"> <li>• Change-data (CD) and unit-of-work (UOW) table pruning information</li> <li>• DB2 for z/OS table space partitioning and compression information</li> <li>• Analysis of target indexes for subscription keys</li> </ul> <b>simple</b> Generates the information in the standard report but excludes subcolumn details.
<b>-tl</b> <i>n</i>	Specifies the date range (0 to 30 days) of entries to be retrieved from the Apply trail (IBMSNAP_APPLYTRAIL) table. The default is 3 days.
<b>-at</b> <i>n</i>	Specifies the date range (0 to 30 days) of entries to be retrieved from the Apply trace (IBMSNAP_APPLYTRACE) table. The default is 3 days.
<b>-ct</b> <i>n</i>	Specifies the date range (0 to 30 days) of entries to be retrieved from the Capture trace (IBMSNAP_CAPTRACE) table. The default is 3 days.
<b>-cm</b> <i>n</i>	Specifies the date range (0 to 30 days) of entries to be retrieved from the Capture monitor (IBMSNAP_CAPMON) table. The default is 3 days.
<b>-sg</b> <i>n</i>	Specifies the date range (0 to 30 days) of entries to be retrieved from the signal (IBMSNAP_SIGNAL) table. The default is 3 days.



Table 19. *asnanalyze* invocation parameter definitions for UNIX and Windows operating systems (continued)

Parameter	Definition
<b>-aq</b> <i>apply_qualifier</i>	Specifies the Apply qualifier that identifies the specific subscription sets to be analyzed.  You can specify more than one Apply qualifier. If there is more than one Apply qualifier, use blank spaces to separate the values. If no Apply qualifier is specified, all subscription sets for the specified database aliases are analyzed.
<b>-cs</b> <i>capture_schema</i>	Specifies the name of the Capture schema that you want to analyze.  If you use this parameter, you can specify only one Capture schema.
<b>-od</b> <i>output_directory</i>	Specifies the directory in which you want to store the Analyzer report. The default is the current directory.
<b>-fn</b> <i>output_filename</i>	Specifies the name of the file that will contain the Analyzer report output.  Use the file naming conventions of the operating system that you are using to run the Analyzer. If the file name already exists, the file is overwritten. The default file name is asnanalyze.htm.
<b>-pw</b> <i>password_filepath</i>	Specifies the name and path of the password file. If you do not specify this parameter, the Analyzer checks the current directory for the asnpwd.aut file.

## Examples for asnanalyze

The following examples illustrate how to use the **asnanalyze** command.

### Example 1

To analyze the replication control tables on a database called proddb1:

```
asnanalyze -db proddb1
```

### Example 2

To obtain a detailed level of analysis about the replication control tables on the proddb1 and proddb2 databases:

```
asnanalyze -db proddb1 proddb2 -la detailed
```

### Example 3

To analyze the last two days of information from the IBMSNAP\_APPLYTRAIL, IBMSNAP\_APPLYTRACE, IBMSNAP\_CAPTRACE, IBMSNAP\_CAPMON, and IBMSNAP\_SIGNAL tables on the proddb1 and proddb2 databases:

## asnanalyze

```
asnanalyze -db proddb1 proddb2 -tl 2 -at 2 -ct 2 -cm 2 -sg 2
```

### Example 4

To obtain a simple level of analysis about the last two days of information from the IBMSNAP\_APPLYTRAIL, IBMSNAP\_APPLYTRACE, IBMSNAP\_CAPTRACE, IBMSNAP\_CAPMON, and IBMSNAP\_SIGNAL tables on the proddb1 and proddb2 databases for only the qual1 and qual2 Apply qualifiers:

```
asnanalyze -db proddb1 proddb2 -la simple -tl 2 -at 2 -ct 2 -cm 2 -sg 2  
-aq qual1 qual2 -od c:\mydir -fn anzout -pw c:\SQLLIB
```

This command example writes the analyzer output to a file named anzout under the c:\mydir directory and uses the password information from the c:\SQLLIB directory.

### Example 5

To analyze a specific Capture schema:

```
asnanalyze -db proddb1 proddb2 -cs BSN
```

### Example 6

To display command help:

```
asnanalyze
```

### Related reference:

- “ANZDPR: Operating the Analyzer (OS/400)” on page 387

---

## asnapply: Starting Apply (UNIX, Windows, z/OS)

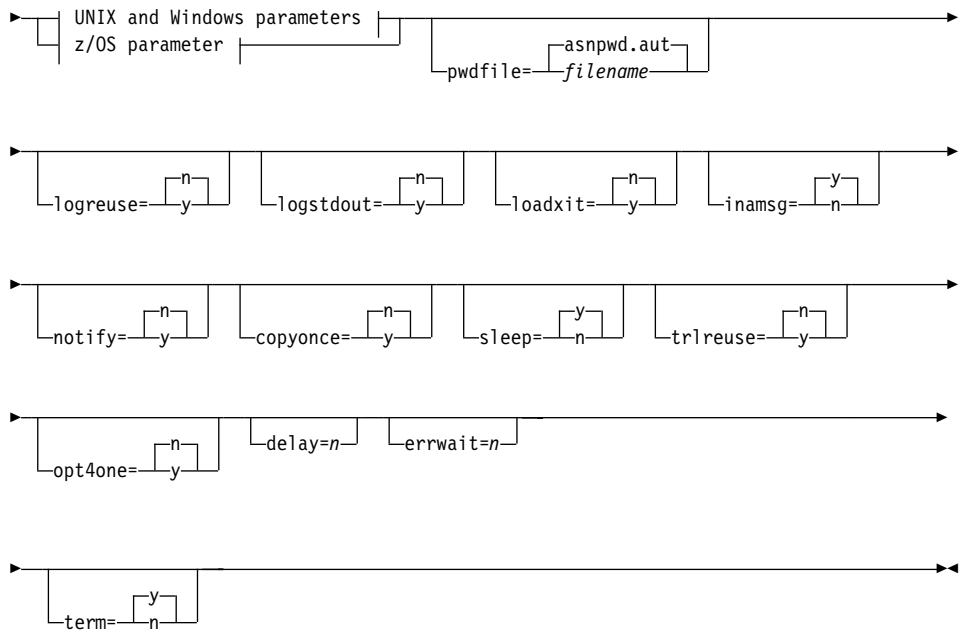
Use the **asnapply** command to start the Apply program on UNIX, Windows, and UNIX System Services (USS) on z/OS. Run this command at an operating system prompt or in a shell script.

After you start the Apply program, it runs continuously until:

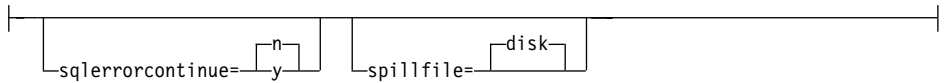
- You stop it in an orderly way.
- You cancel it.
- An unexpected error or failure occurs.

### To start the Apply program using the *asnapply* command:

```
➤—asnapply—apply_qual=apply_qualifier—db2_subsystem=name—(1)➤  
└─control_server=db_name┘└─apply_path=pathname┘➤
```



#### UNIX and Windows parameters:



#### z/OS parameter:



#### Notes:

- 1 Use the `db2_subsystem` parameter only on z/OS operating systems.

Table 20 on page 310 defines the invocation parameters.

Table 20. *asnapply* invocation parameter definitions for UNIX, Windows, and z/OS operating systems

Parameter	Definition
<b>apply_qual=apply_qualifier</b>	<p>Specifies the Apply qualifier that the Apply program uses to identify the subscription sets to be served. This parameter is required.</p> <p>The value that you enter must match the value of the APPLY_QUAL column in the subscription sets (IBMSNAP_SUBS_SET) table. The Apply qualifier name is case sensitive and can be a maximum of 18 characters.</p>
<b>db2_subsystem=name</b>	<p><b>For z/OS only:</b> Specifies the name of the DB2 subsystem. The DB2 subsystem name that you enter can be a maximum of four characters. There is no default for this parameter. This parameter is required.</p>
<b>control_server=db_name</b>	<p>Specifies the name of the Apply control server on which the subscription definitions and Apply program control tables reside.</p> <p><b>For UNIX and Windows:</b> If you do not specify an Apply control server, this parameter defaults to the value from the DB2DBDFT environment variable.</p> <p><b>For z/OS:</b> The control server parameter is the name of the database server that connects to the control server.</p>
<b>apply_path=pathname</b>	<p>Specifies the location of the work files used by the Apply program. The default is the directory where the <b>asnapply</b> command was invoked.</p>
<b>pwdfile=filename</b>	<p>Specifies the name of the password file. If you do not specify a password file, the default is asnpwd.aut.</p> <p>This command searches for the password file in the directory specified by the <b>apply_path</b> parameter. If no <b>apply_path</b> parameter is specified, this command searches for the password file in the directory where the command was invoked.</p>

Table 20. *asnapply* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>logreuse=y/n</b>	<p>Specifies whether the Apply program reuses or appends messages to the log file (<i>db2instance.control_server.apply_qualifier.APP.log</i>).</p> <p><b>n</b> (default) The Apply program appends messages to the log file, even after the Apply program is restarted.</p> <p><b>y</b> The Apply program reuses the log file by deleting it and then re-creating it when the Apply program is restarted.</p> <p><b>For z/OS:</b> The log file name does not contain the DB2 instance name (<i>control_server.apply_qualifier.APP.log</i>).</p>
<b>logstdout=y/n</b>	<p>Specifies where messages are sent by the Apply program.</p> <p><b>n</b> (default) The Apply program sends messages to the log file only.</p> <p><b>y</b> The Apply program sends messages to both the log file and the standard output (stdout).</p>
<b>loadxit=y/n</b>	<p>Specifies whether the Apply program invokes ASNLOAD. ASNLOAD is an IBM-supplied exit routine that uses the export and load utilities to refresh target tables.</p> <p><b>n</b> (default) The Apply program does not invoke ASNLOAD.</p> <p><b>y</b> The Apply program invokes ASNLOAD.</p>
<b>inamsg=y/n</b>	<p>Specifies whether the Apply program issues a message when the Apply program is inactive.</p> <p><b>y</b> (default) The Apply program issues a message when inactive.</p> <p><b>n</b> The Apply program does not issue a message when inactive.</p>

Table 20. *asnapply* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>notify=y/n</b>	<p>Specifies whether the Apply program should invoke ASNDONE. ASNDONE is an exit routine that returns control to you when the Apply program finishes copying a subscription set.</p> <p><b>n</b> (default) The Apply program does not invoke ASNDONE.</p> <p><b>y</b> The Apply program invokes ASNDONE.</p>
<b>copyonce=y/n</b>	<p>Specifies whether the Apply program executes one copy cycle for each subscription set that is eligible at the time the Apply program is invoked. Then the Apply program terminates. An eligible subscription set meets the following criteria:</p> <ul style="list-style-type: none"> <li>• (ACTIVATE &gt; 0) in the subscription sets (IBMSNAP_SUBS_SET) table. When the ACTIVATE column value is greater than zero, the subscription set is active indefinitely or is used for a one-time-only subscription processing.</li> <li>• (REFRESH_TYPE = R or B) or (REFRESH_TYPE = E and the specified event occurred). The REFRESH_TYPE column value is stored in the IBMSNAP_SUBS_SET table.</li> </ul> <p>The MAX_SYNCH_MINUTES limit from the subscription sets table and the END_OF_PERIOD timestamp from the subscription events (IBMSNAP_SUBS_EVENT) table are honored if specified.</p> <p><b>n</b> (default) The Apply program does not execute one copy cycle for each eligible subscription set.</p> <p><b>y</b> The Apply program executes one copy cycle for each eligible subscription set.</p>
<b>sleep=y/n</b>	<p>Specifies how the Apply program is to proceed if no new subscription sets are eligible for processing.</p> <p><b>y</b> (default) The Apply program goes to sleep.</p> <p><b>n</b> The Apply program stops.</p>

Table 20. *asnapply* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>trlreuse=y/n</b>	<p>Specifies whether the Apply program empties the Apply trail (IBMSNAP_APPLYTRAIL) table when the Apply program starts.</p> <p><b>n</b> (default) The Apply program appends entries to the IBMSNAP_APPLYTRAIL table. The Apply program does not empty the table.</p> <p><b>y</b> The Apply program empties the IBMSNAP_APPLYTRAIL table during program startup.</p>
<b>opt4one=y/n</b>	<p>Specifies whether the performance of the Apply program is optimized if only one subscription set is defined for the Apply program.</p> <p><b>n</b> (default) The performance of the Apply program is not optimized for one subscription set.</p> <p><b>y</b> The performance of the Apply program is optimized for one subscription set.</p> <p>If you set optimization to y, the Apply program caches and reuses the information about the subscription-set members. This reuse of subscription-set member information reduces CPU usage and improves throughput rates.</p>
<b>delay=n</b>	<p>Specifies the delay time (in seconds) at the end of each Apply cycle when continuous replication is used, where <math>n=0, 1, 2, 3, 4, 5</math>, or 6. The default is 6.</p>
<b>errwait=n</b>	<p>Specifies the number of seconds (1 to 300) that the Apply program waits before retrying after the program encounters an error condition. The default value is 300 seconds (five minutes).</p> <p><b>Important:</b> Do not specify too small a number, because the Apply program runs almost continuously and generates many rows in the Apply trail (IBMSNAP_APPLYTRAIL) table.</p>

Table 20. *asnapply* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>term=y/n</b>	<p>Specifies how the status of DB2 affects the operation of the Apply program.</p> <p><b>y</b> (default) The Apply program terminates if DB2 terminates.</p> <p><b>n</b> The Apply program waits for DB2 to start, if DB2 is not active.</p> <p><b>For UNIX and Windows:</b> If DB2 quiesces and the Apply program is active, the Apply program stays active and does not reconnect until DB2 is out of quiesce mode.</p> <p><b>For z/OS:</b> If DB2 quiesces and the Apply program is active, the Apply program stays active and does not reconnect until DB2 is started again.</p>
<b>sqlerrorcontinue=y/n</b>	<p><b>For UNIX and Windows only:</b> Specifies whether the Apply program continues processing when it encounters certain SQL errors.</p> <p>The Apply program checks the failing SQLSTATE against the values specified in the SQLSTATE file, which you create before running the Apply program. If a match is found, the Apply program writes the information about the failing row to an error file (<i>apply_qualifier.ERR</i>) and continues processing. The SQLSTATE file can contain up to 20 five-byte values.</p> <p><b>n</b> (default) The Apply program does not check the SQLSTATE file.</p> <p><b>y</b> The Apply program checks the SQLSTATE file during processing.</p>



Table 20. *asnapply* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>spillfile=</b> <i>filetype</i>	Specifies where the fetched answer set is stored.  <b>For UNIX and Windows</b> , valid values are:  <b>disk</b> (default) A disk file.  <b>For z/OS</b> , valid values are:  <b>mem</b> (default) A memory file. The Apply program fails if there is insufficient memory for the answer set.  <b>disk</b> A disk file.

## Examples for **asnapply**

The following examples illustrate how to use the **asnapply** command.

### Example 1

To start an Apply program using an Apply qualifier named AQ1, a control server named dbx with work files located in the /home/files/apply/ directory:

```
asnapply apply_qual=AQ1 control_server=dbx apply_path=/home/files/apply/
pwdfile=pass1.txt
```

The Apply program searches the /home/files/apply/ directory for the password file named pass1.txt.

### Example 2

To start an Apply program that invokes the ASNLOAD exit routine:

```
asnapply apply_qual=AQ1 control_server=dbx pwdfile=pass1.txt loadxit=y
```

In this example, the Apply program searches the current directory for the password file named pass1.txt.

### Example 3

To start an Apply program that executes one copy cycle for each eligible subscription set:

```
asnapply apply_qual=AQ1 control_server=dbx apply_path=/home/files/apply/
copyonce=y
```

In this example, the Apply program searches the /home/files/apply/ directory for the default password file named asnpwd.aut.

**Related tasks:**

- Chapter 19, “Operating the replication programs (z/OS)” on page 453

**Related reference:**

- “STRDPRAPY: Starting Apply (OS/400)” on page 427

**asnscap: Starting Capture (UNIX, Windows, z/OS)**

Use the **asnscap** command to start the Capture program on UNIX, Windows, and UNIX System Services (USS) on z/OS. Run this command at an operating system prompt or in a shell script rather than through the Replication Center.

After you start the Capture program, it runs continuously until you stop it or it detects an unrecoverable error.

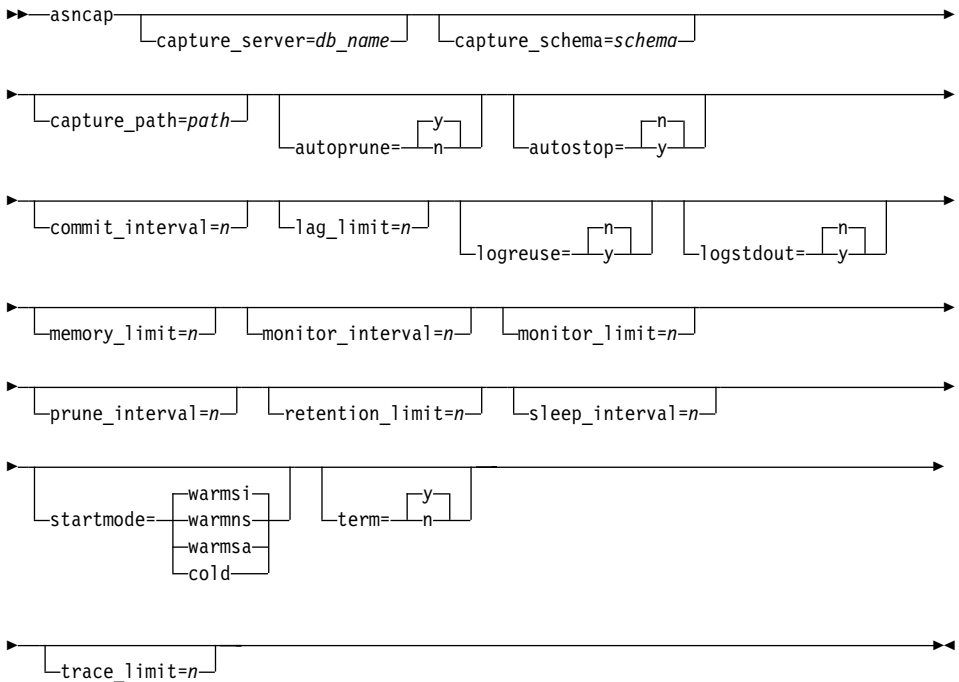
***To start the Capture program using the asncap command:***

Table 21 defines the invocation parameters.

Table 21. *asncap* invocation parameter definitions for UNIX, Windows, and z/OS operating systems

Parameter	Definition
<b>capture_server=db_name</b>	<p>Specifies the name of the Capture control server.</p> <p><b>For UNIX and Windows:</b> If you do not specify a Capture control server, this parameter defaults to the value from the DB2DBDFT environment variable.</p> <p><b>For z/OS:</b> The <b>capture_server</b> parameter is the name of the database server that connects to the control server. For data sharing, do not use the group attach name. Instead, specify a member subsystem name.</p>
<b>capture_schema=schema</b>	<p>Specifies the name of the Capture schema that is used to identify a particular Capture program. The schema name that you enter must be 1 to 30 characters in length. The default is ASN.</p>
<b>capture_path=path</b>	<p>Specifies the location of the work files used by the Capture program. The default is the directory where the <b>asncap</b> command was invoked.</p>
<b>autoprune=y/n</b>	<p>Specifies whether automatic pruning of the rows in the change-data (CD), unit-of-work (UOW), Capture monitor (IBMSNAP_CAPMON), Capture trace (IBMSNAP_CAPTRACE), and signal (IBMSNAP_SIGNAL) tables is enabled.</p> <p><b>y</b> (default) The Capture program automatically prunes the eligible rows at the interval specified in the Capture parameters (IBMSNAP_CAPPARMS) table. The Capture program prunes the CD, UOW, and IBMSNAP_SIGNAL rows that are older than the retention limit, regardless of whether the rows have been replicated.</p> <p><b>n</b> Automatic pruning is disabled.</p>
<b>autostop=y/n</b>	<p>Specifies whether the Capture program terminates after retrieving all the transactions that were logged before the Capture program started.</p> <p><b>n</b> (default) The Capture program does not terminate after retrieving the transactions.</p> <p><b>y</b> The Capture program terminates after retrieving the transactions.</p>

Table 21. *asncap* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>commit_interval=n</b>	Specifies the number of seconds that the Capture program waits before committing rows to the unit-of-work (UOW) and change-data (CD) tables. The default is 30 seconds.
<b>lag_limit=n</b>	Specifies the number of minutes that the Capture program is allowed to lag in processing log records before shutting down. The default is 10 080 minutes (seven days).
<b>logreuse=y/n</b>	<p>Specifies whether the Capture program reuses or appends messages to the log file (<i>db2instance.capture_server.capture_schema.CAP.log</i>).</p> <p><b>n</b> (default) The Capture program appends messages to the log file, even after the Capture program is restarted.</p> <p><b>y</b> The Capture program reuses the log file by first truncating the current log file and then starting a new log when the Capture program is restarted.</p> <p><b>For z/OS:</b> The log file name does not contain the DB2 instance name (<i>capture_server.capture_schema.CAP.log</i>).</p>
<b>logstdout=y/n</b>	<p>Specifies where messages are sent by the Capture program.</p> <p><b>n</b> (default) The Capture program sends messages to the log file only.</p> <p><b>y</b> The Capture program sends messages to both the log file and the standard output (stdout).</p>
<b>memory_limit=n</b>	Specifies the maximum size (in megabytes) of memory that the Capture program can use to build transactions. After reaching this memory limit, the Capture program spills transactions to a file. The default is 32 megabytes.
<b>monitor_interval=n</b>	Specifies how frequently (in seconds) the Capture program inserts rows into the Capture monitor (IBMSNAP_CAPMON) table. The default is 300 seconds (five minutes).
<b>monitor_limit=n</b>	Specifies how long (in minutes) a row can remain in the Capture monitor (IBMSNAP_CAPMON) table before it becomes eligible for pruning. All IBMSNAP_CAPMON rows that are older than the value of the <b>monitor_limit</b> parameter are pruned at the next pruning cycle. The default is 10 080 minutes (seven days).

Table 21. *asncap* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>prune_interval</b> = <i>n</i>	Specifies how frequently (in seconds) the change-data (CD), unit-of-work (UOW), Capture monitor (IBMSNAP_CAPMON), Capture trace (IBMSNAP_CAPTRACE), and signal (IBMSNAP_SIGNAL) tables are pruned. This parameter is ignored if you set the autoprune parameter to <i>n</i> . The default is 300 seconds (five minutes).
<b>retention_limit</b> = <i>n</i>	Specifies how long (in minutes) a row can remain in the change-data (CD), unit-of-work (UOW), or signal (IBMSNAP_SIGNAL) table before it becomes eligible for pruning. Each row that is older than the value of the <b>retention_limit</b> parameter is pruned at the next pruning cycle. The default is 10 080 minutes (seven days).
<b>sleep_interval</b> = <i>n</i>	<p>Specifies the number of seconds that the Capture program sleeps when it finishes processing the active log and determines that the buffer is empty. The default is five seconds.</p> <p><b>For z/OS:</b> Specifies the number of seconds that the Capture program sleeps after the buffer returns less than half full.</p>

Table 21. *asncap* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>startmode=mode</b>	<p>Specifies the processing procedure used by the Capture program during a Capture startup.</p> <p><b>warmsi</b> (default)</p> <p>The Capture program resumes processing where it ended in its previous run if warm start information is available. If this is the first time that you are starting the Capture program, it automatically switches to a cold start.</p> <p>During a warm start, the Capture program leaves the Capture trace (IBMSNAP_CAPTRACE), change-data (CD), unit-of-work (UOW), and restart (IBMSNAP_RESTART) tables intact. If errors occur after the Capture program started, the Capture program terminates.</p> <p><b>warmns</b></p> <p>The Capture program resumes processing where it ended in its previous run if warm start information is available. If errors occur after the Capture program started, the Capture program terminates. If the Capture program cannot warm start, it <i>does not</i> switch to a cold start.</p> <p><b>warmsa</b></p> <p>The Capture program resumes processing where it ended in its previous run if warm start information is available. If the Capture program cannot warm start, it switches to a cold start.</p> <p><b>cold</b></p> <p>The Capture program starts by deleting all rows in its CD, UOW, and IBMSNAP_CAPTRACE tables during initialization. All subscriptions to replication sources are fully refreshed during the next Apply processing cycle. A full refresh is not done if the target is a noncomplete consistent-change data (CCD) table.</p>

Table 21. *asncap* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>term=y/n</b>	<p>Specifies whether the Capture program terminates if DB2 terminates.</p> <p><b>y</b> (default) The Capture program terminates if DB2 terminates.</p> <p><b>n</b> The Capture program continues running if DB2 terminates with MODE(QUIESCE). When DB2 initializes, the Capture program starts in warm mode and begins capturing at the point it left off when DB2 terminated.</p> <p>If DB2 terminates via FORCE or due to abnormal termination, the Capture program terminates even if you set this parameter to n.</p> <p>If you set this parameter to n and start DB2 with restricted access (ACCESS MAINT), the Capture program cannot connect and subsequently terminates.</p>
<b>trace_limit=n</b>	<p>Specifies how long (in minutes) a row can remain in the Capture trace (IBMSNAP_CAPTRACE) table before it becomes eligible for pruning. All IBMSNAP_CAPTRACE rows that are older than the value of the <b>trace_limit</b> parameter are pruned at the next pruning cycle. The default is 10 080 minutes (seven days).</p>

## Examples for asncap

The following examples illustrate how to use the **asncap** command.

### Example 1

To start a Capture program for the first time using a Capture control server named db and a Capture schema of ASN with work files located in the /home/files/capture/logs/ directory:

```
asncap capture_server=db capture_schema=ASN
      capture_path=/home/files/capture/logs/ startmode=cold
```

### Example 2

To restart a Capture program without pruning after the Capture program was stopped:

```
asncap capture_server=db autoprune=n sleep_interval=10 startmode=warmsa
```

The Capture program in this example retains all rows in the corresponding control tables and sleeps for ten seconds after it finishes processing the active

log and determines that the buffer is empty. The Capture program resumes processing where it ended in its previous run and switches to a cold start if warm start information is unavailable.

### Example 3

To restart a Capture program with the warmns startmode and changed parameter settings:

```
asncap capture_server=db autoprun=y prune_interval=60 retention_limit=1440
startmode=warmns
```

This command restarts the Capture program and uses new parameter settings to decrease the amount of time before the CD, UOW, and IBMSNAP\_SIGNAL tables become eligible for pruning and to increase the frequency of pruning from the default parameter settings. The Capture program resumes processing where it ended in its previous run but does not automatically switch to a cold start if warm start information is unavailable.

### Example 4

To start a Capture program that sends all of its work files to a new subdirectory called capture\_files:

1. Go to the appropriate directory, and then create a new subdirectory called capture\_files:

```
cd /home/db2inst
mkdir capture_files
```

2. Start the Capture program, and specify a Capture path that is located in the new subdirectory that you just created:

```
asncap capture_server=db capture_schema=ASN
capture_path=/home/db2inst/capture_files startmode=warmsi
```

### Related tasks:

- Chapter 19, “Operating the replication programs (z/OS)” on page 453

### Related reference:

- “STRDPRCAP: Starting Capture (OS/400)” on page 436

---

## asnccmd: Operating Capture (UNIX, Windows, z/OS)

Use the **asnccmd** command to operate the Capture program on UNIX, Windows, and UNIX System Services (USS) on z/OS. Run this command at an operating system prompt or in a shell script.

### *To operate the Capture program using the asnccmd command:*

```
→asnccmd [capture_server=db_name] [capture_schema=schema]
```





Parameters:

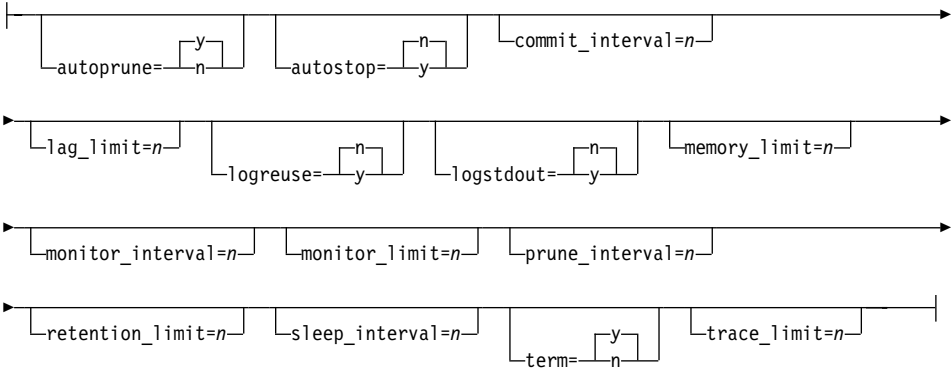


Table 22 defines the invocation parameters.

Table 22. *asnccmd* invocation parameter definitions for UNIX, Windows, and z/OS operating systems

Parameter	Definition
<b>capture_server=db_name</b>	<p>Specifies the name of the Capture control server.</p> <p><b>For UNIX and Windows:</b> If you do not specify a Capture control server, this parameter defaults to the value from the DB2DBDFT environment variable.</p> <p><b>For z/OS:</b> The name of the database server that connects to the control server. For data sharing, do not use the group attach name. Instead, specify a member subsystem name.</p>
<b>capture_schema=schema</b>	<p>Specifies the name of the Capture schema that is used to identify a particular Capture program. The schema name that you enter must be 1 to 30 characters in length. The default is ASN.</p>

Table 22. *asnccmd* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>chgparms</b>	Specify to change the parameter values of a Capture program while it is running. You can specify new parameter values or override the values that were passed to the Capture program when it started. To determine which parameters can be overridden, see Table 23 on page 325.
<b>prune</b>	Specify this parameter if you want to prune the change-data (CD), unit-of-work (UOW), Capture monitor (IBMSNAP_CAPMON), Capture trace (IBMSNAP_CAPTRACE), and signal (IBMSNAP_SIGNAL) tables once. The Capture program issues a message when the command is successfully queued.
<b>qryparms</b>	Specify if you want the current operational parameter values written to the standard output (stdout).
<b>reinit</b>	<p>Specify to have the Capture program obtain newly added replication sources from the register (IBMSNAP_REGISTER) table or new tuning parameters from the Capture parameters (IBMSNAP_CAPPARMS) table. For example, use this parameter if you add a new replication source or if you use ALTER ADD to add a column to a replication source and change-data (CD) table while the Capture program is running.</p> <p><b>Important:</b> Do not use the reinit parameter to reinitialize the Capture program after canceling a replication source or dropping a replication source table while the Capture program is running. Instead, stop the Capture program and restart it using the <b>asncap</b> command with the startmode parameter set to warmsa, warmns, or warmsi.</p>
<b>resume</b>	Specify to have a suspended Capture program resume capturing data.
<b>status</b>	Specify to receive messages that indicate the state of each Capture thread (administration, pruning, serialization, and worker).
<b>stop</b>	Specify to stop the Capture program in an orderly way and commit the log records processed up to that point.
<b>suspend</b>	<p>Specify to relinquish operating system resources to operational transactions during peak periods without destroying the Capture program environment.</p> <p><b>Important:</b> Do not suspend Capture to cancel a replication source. Instead, stop the Capture program.</p>

Table 23 defines the chgparms invocation parameters.

*Table 23. asnccmd chgparms parameter definitions for UNIX, Windows, and z/OS operating systems*

Parameter	Definition
<b>autoprune=y/n</b>	<p>Specifies whether automatic pruning of the rows in the change-data (CD), unit-of-work (UOW), Capture monitor (IBMSNAP_CAPMON), Capture trace (IBMSNAP_CAPTRACE), and signal (IBMSNAP_SIGNAL) tables is enabled.</p> <p><b>y</b> (default) The Capture program automatically prunes the eligible rows at the interval specified in the Capture parameters (IBMSNAP_CAPPARMS) table. The Capture program prunes the CD, UOW, and IBMSNAP_SIGNAL rows that are older than the retention limit, regardless of whether the rows have been replicated.</p> <p><b>n</b> Automatic pruning is disabled.</p>
<b>autostop=y/n</b>	<p>Specifies whether the Capture program terminates after retrieving all the transactions logged before the Capture program started.</p> <p><b>n</b> (default) The Capture program does not terminate after retrieving the transactions.</p> <p><b>y</b> The Capture program terminates after retrieving the transactions.</p>
<b>commit_interval=n</b>	<p>Specifies the number of seconds that the Capture program waits before committing rows to the unit-of-work (UOW) and change-data (CD) tables. The default is 30 seconds.</p>
<b>lag_limit=n</b>	<p>Specifies the number of minutes that the Capture program is allowed to lag in processing log records before shutting down. The default is 10 080 minutes (seven days).</p>

Table 23. *asnccmd chgparms* parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>logreuse=y/n</b>	<p>Specifies whether the Capture program reuses or appends messages to the log file (<i>db2instance.capture_server.capture_schema.CAP.log</i>).</p> <p><b>n</b> (default) The Capture program appends messages to the log file, even after the Capture program is restarted.</p> <p><b>y</b> The Capture program reuses the log file by first truncating the current log file and then starting a new log when the Capture program is restarted.</p> <p>If you change this parameter to <b>y</b> through the use of the <b>chgparms</b> parameter, the log is immediately truncated and reused. This change to the <b>logreuse</b> parameter does not affect the next start of the Capture program.</p> <p><b>For z/OS:</b> The log file name does not contain the DB2 instance name (<i>capture_server.capture_schema.CAP.log</i>).</p>
<b>logstdout=y/n</b>	<p>Specifies where messages are sent by the Capture program.</p> <p><b>n</b> (default) The Capture program sends messages to the log file only.</p> <p><b>y</b> The Capture program sends messages to both the log file and the standard output (stdout).</p>
<b>memory_limit=n</b>	<p>Specifies the maximum size (in megabytes) of memory that the Capture program can use to build transactions. After reaching this memory limit, the Capture program spills transactions to a file. The default is 32 megabytes.</p>
<b>monitor_interval=n</b>	<p>Specifies how frequently (in seconds) the Capture program inserts rows into the Capture monitor (IBMSNAP_CAPMON) table. The default is 300 seconds (five minutes).</p>
<b>monitor_limit=n</b>	<p>Specifies how long (in minutes) a row can remain in the Capture monitor (IBMSNAP_CAPMON) table before it becomes eligible for pruning. All IBMSNAP_CAPMON rows that are older than the value of the <b>monitor_limit</b> parameter are pruned at the next pruning cycle. The default is 10 080 minutes (seven days).</p>

Table 23. *asnccmd chgparms* parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>prune_interval</b> = <i>n</i>	Specifies how frequently (in seconds) the change-data (CD), unit-of-work (UOW), Capture monitor (IBMSNAP_CAPMON), Capture trace (IBMSNAP_CAPTRACE), and signal (IBMSNAP_SIGNAL) tables are pruned. This parameter is ignored if you set the autoprune parameter to <i>n</i> . The default is 300 seconds (five minutes).
<b>retention_limit</b> = <i>n</i>	Specifies how long (in minutes) a row can remain in the change-data (CD), unit-of-work (UOW), or signal (IBMSNAP_SIGNAL) table before it becomes eligible for pruning. Each row that is older than the value of the <b>retention_limit</b> parameter is pruned at the next pruning cycle. The default is 10 080 minutes (seven days).
<b>sleep_interval</b> = <i>n</i>	<p>Specifies the number of seconds that the Capture program sleeps when it finishes processing the active log and determines that the buffer is empty. The default is five seconds.</p> <p><b>For z/OS:</b> Specifies the number of seconds that the Capture program sleeps after the buffer returns less than half full.</p>
<b>term</b> = <i>y/n</i>	<p>Specifies whether the Capture program terminates if DB2 terminates.</p> <p><b>y (default)</b></p> <p>The Capture program terminates if DB2 terminates.</p> <p><b>n</b></p> <p>The Capture program continues running if DB2 terminates with MODE(QUIESCE). When DB2 initializes, the Capture program starts in warm mode and begins capturing at the point it left off when DB2 terminated.</p> <p>If DB2 terminates via FORCE or due to an abnormal termination, the Capture program terminates even if you set this parameter to <i>n</i>.</p> <p>If you set this parameter to <i>n</i> and start DB2 with restricted access (ACCESS MAINT), the Capture program cannot connect and subsequently terminates.</p>

Table 23. *asnccmd chgparms* parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>trace_limit=<i>n</i></b>	Specifies how long (in minutes) a row can remain in the Capture trace (IBMSNAP_CAPTRACE) table before it becomes eligible for pruning. All IBMSNAP_CAPTRACE rows that are older than the value of the <b>trace_limit</b> parameter are pruned at the next pruning cycle. The default is 10 080 minutes (seven days).

Examples for **asnccmd**

The following examples illustrate how to use the **asnccmd** command.

**Example 1**

To enable a running Capture program to recognize newly added replication sources:

```
asnccmd capture_server=db capture_schema=ASN reinit
```

**Example 2**

To prune the CD, UOW, IBMSNAP\_CAPMON, IBMSNAP\_CAPTRACE, and IBMSNAP\_SIGNAL tables once:

```
asnccmd capture_server=db capture_schema=ASN prune
```

**Example 3**

To receive messages about the state of each Capture thread:

```
asnccmd capture_server=db capture_schema=ASN status
```

**Example 4**

To send the current operational values of a Capture program to the standard output:

```
asnccmd capture_server=db capture_schema=ASN qryparms
```

**Example 5**

To disable the automatic pruning in a running Capture program:

```
asnccmd capture_server=db capture_schema=ASN chgparms autoprunen=n
```

**Example 6**

To stop a running Capture program:

```
asnccmd capture_server=db capture_schema=ASN stop
```

**Related tasks:**

- Chapter 19, “Operating the replication programs (z/OS)” on page 453

**Related reference:**

- “OVRDPRCAPA: Overriding DPR capture attributes (OS/400)” on page 414

## asnmcmd: Operating the Replication Alert Monitor (UNIX, Windows, z/OS)

Use the **asnmcmd** command to operate the Replication Alert Monitor on UNIX, Windows, and UNIX System Services (USS) on z/OS. Run this command at an operating system prompt or in a shell script.

**To operate the Replication Alert Monitor using the *asnmcmd* command:**

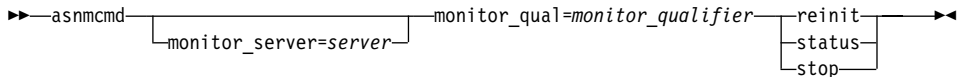


Table 24 defines the invocation parameters.

*Table 24. asnmcmd invocation parameter definitions for UNIX, Windows, and z/OS operating systems*

Parameter	Definition
<b>monitor_server=server</b>	<p>Specifies the name of the monitor control server where the Replication Alert Monitor program runs and the monitor control tables reside. This must be the first parameter if entered.</p> <p><b>For UNIX and Windows:</b> If you do not specify a monitor control server, this parameter defaults to the value from the DB2DBDFT environment variable.</p> <p><b>For z/OS:</b> The default is DSN.</p>
<b>monitor_qual=monitor_qualifier</b>	<p>Specifies the monitor qualifier that the Replication Alert Monitor program uses. The monitor qualifier identifies the server to be monitored and the associated monitoring conditions.</p> <p>You must specify a monitor qualifier. The monitor qualifier name is case sensitive and can be a maximum of 18 characters.</p>
<b>reinit</b>	Specify to have the Replication Alert Monitor program obtain new parameters from the monitor control tables.
<b>status</b>	Specify to receive messages that indicate the state of each thread (administration, serialization, and worker) in the Replication Alert Monitor.
<b>stop</b>	Specify to stop the Replication Alert Monitor in an orderly way.

Examples for asnmcmd

The following examples illustrate how to use the **asnmcmd** command.

Example 1

To stop the Replication Alert Monitor for the specified monitor qualifier:

```
asnmcmd monitor_server=wsdb monitor_qual=monqual stop
```

Example 2

To receive messages that indicate the status of the Replication Alert Monitor threads:

```
asnmcmd monitor_server=wsdb monitor_qual=monqual status
```

Example 3

To refresh the Replication Alert Monitor with current values from the monitor control tables:

```
asnmcmd monitor_server=wsdb monitor_qual=monqual reinit
```

Related tasks:

- Chapter 11, “Monitoring replication” on page 161

Related reference:

- “asnmon: Starting the Replication Alert Monitor (UNIX, Windows, z/OS)” on page 330

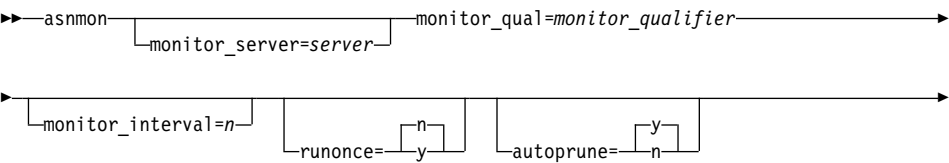
asnmon: Starting the Replication Alert Monitor (UNIX, Windows, z/OS)

Use the **asnmon** command to start the Replication Alert Monitor on UNIX, Windows, and UNIX System Services (USS) on z/OS. Run this command at an operating system prompt or in a shell script.

The Replication Alert Monitor records the following information:

- The status of Capture and Apply programs
- Capture and Apply error messages written to the control tables
- Threshold values

To start the Replication Alert Monitor using the *asnmon* command:





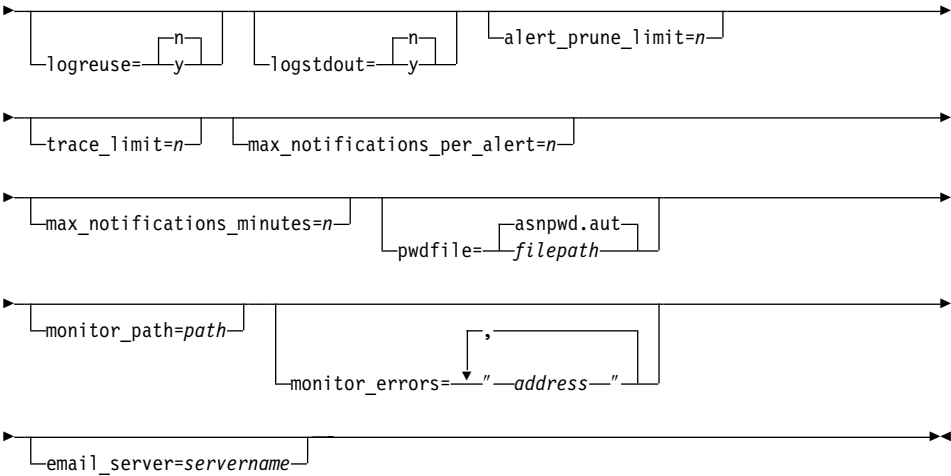


Table 25 defines the invocation parameters.

Table 25. *asnmon* invocation parameter definitions for UNIX, Windows, and z/OS operating systems

Parameter	Definition
<b>monitor_server=server</b>	<p>Specifies the name of the monitor control server where the Replication Alert Monitor program runs and the monitor control tables reside. This must be the first parameter if entered.</p> <p><b>For UNIX and Windows:</b> If you do not specify a monitor control server, this parameter defaults to the value from the DB2DBDFT environment variable.</p> <p><b>For z/OS:</b> The default is DSN.</p>
<b>monitor_qual=monitor_qualifier</b>	<p>Specifies the monitor qualifier that the Replication Alert Monitor program uses. The monitor qualifier identifies the server to be monitored and the associated monitoring conditions.</p> <p>You must specify a monitor qualifier. The monitor qualifier name is case sensitive and can be a maximum of 18 characters.</p>

Table 25. *asnmon* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>monitor_interval</b> = <i>n</i>	<p>Specifies how frequently (in seconds) the Replication Alert Monitor program runs for this monitor qualifier. The default is 300 seconds (five minutes).</p> <p>This parameter is ignored by the Replication Alert Monitor if you set the <b>runonce</b> parameter to <b>y</b>.</p> <p><b>Important:</b> This <b>monitor_interval</b> parameter affects the Replication Alert Monitor program only; this parameter does not affect Capture programs.</p>
<b>runonce</b> = <i>y/n</i>	<p>Specifies whether the Replication Alert Monitor program runs only one time for this monitor qualifier.</p> <p><b>n</b> (default) The Replication Alert Monitor program runs at the frequency indicated by the <b>monitor_interval</b> parameter.</p> <p><b>y</b> The Replication Alert Monitor program runs only one monitor cycle.</p> <p>If you set the <b>runonce</b> parameter to <b>y</b>, the <b>monitor_interval</b> parameter is ignored by the Replication Alert Monitor.</p>
<b>autoprune</b> = <i>y/n</i>	<p>Specifies whether automatic pruning of the rows in the Replication Alert Monitor alerts (IBMSNAP_ALERTS) table is enabled.</p> <p><b>y</b> (default) The Replication Alert Monitor program automatically prunes the rows in the IBMSNAP_ALERTS table that are older than the value of the <b>alert_prune_limit</b> parameter.</p> <p><b>n</b> Automatic pruning is disabled.</p>
<b>logreuse</b> = <i>y/n</i>	<p>Specifies whether the Replication Alert Monitor program reuses or appends messages to the log file (<i>db2instance.monitor_server.monitor_qualifier.MON.log</i>).</p> <p><b>n</b> (default) The Replication Alert Monitor program appends messages to the log file.</p> <p><b>y</b> The Replication Alert Monitor program reuses the log file by deleting it and then re-creating it when the Replication Alert Monitor program is restarted.</p>

Table 25. *asnmon* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>logstdout=y/n</b>	<p>Specifies where messages are sent by the Replication Alert Monitor program.</p> <p><b>n</b> (default) The Replication Alert Monitor program sends messages to the log file only.</p> <p><b>y</b> The Replication Alert Monitor program sends messages to both the log file and the standard output (stdout).</p>
<b>alert_prune_limit=n</b>	Specifies how long (in minutes) rows are kept in the Replication Alert Monitor alerts (IBMSNAP_ALERTS) table. Any rows older than this value are pruned. The default is 10 080 minutes (seven days).
<b>trace_limit=n</b>	Specifies how long (in minutes) a row can remain in the Replication Alert Monitor trace (IBMSNAP_MONTRACE) table before it becomes eligible for pruning. All IBMSNAP_MONTRACE rows that are older than the value of this <b>trace_limit</b> parameter are pruned at the next pruning cycle. The default is 10 080 minutes (seven days).
<b>max_notifications_per_alert=n</b>	Specifies the maximum number of the same alerts that are sent to a user when the alerts occurred during the time period specified by the <b>max_notifications_minutes</b> parameter value. Use this parameter to avoid re-sending the same alerts to a user. The default is 3.
<b>max_notifications_minutes=n</b>	This parameter works with the <b>max_notifications_per_alert</b> parameter to indicate the time period when alert conditions occurred. The default is 60 minutes.
<b>pwdfile=filepath</b>	Specifies the fully qualified name of the password file. You define this file using the <b>asnpwd</b> command. The default file name is asnpwd.aut.
<b>monitor_path=path</b>	Specifies the location of the log files used by the Replication Alert Monitor program. The default is the directory where the <b>asnmon</b> command was invoked.

Table 25. *asnmon* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>monitor_errors=address</b>	<p>Specifies the e-mail address to which notifications are sent if a fatal error is detected before the alert monitor connects to the monitor control server. Use this parameter to send a notification that the monitor control server connection failed because of invalid start parameters, an incorrect monitor qualifier, a down database, or other error.</p> <p>Type double quotation marks around the e-mail address text.</p> <p>You can enter multiple e-mail addresses. Separate the e-mail addresses with commas. You can type spaces before or after the commas.</p>
<b>email_server=servername</b>	<p>Specifies the e-mail server address. Enter this parameter <i>only</i> if you use the ASNMAIL exit routine with SMTP (Simple Mail Transfer Protocol).</p>

Examples for **asnmon**

The following examples illustrate how to use the **asnmon** command.

**Example 1**

To start the Replication Alert Monitor with the default parameters:

```
asnmon monitor_server=wsdb monitor_qual=monqual
```

**Example 2**

To start a Replication Alert Monitor that runs every 120 seconds (two minutes) for the specified monitor qualifier:

```
asnmon monitor_server=wsdb monitor_qual=monqual monitor_interval=120
```

**Example 3**

To start the Replication Alert Monitor and specify that it run only once for the specified monitor qualifier:

```
asnmon monitor_server=wsdb monitor_qual=monqual runonce=y
```

**Example 4**

To start a Replication Alert Monitor that sends e-mail notifications if it detects monitoring errors:

```
asnmon monitor_server=wsdb monitor_qual=monqual
  monitor_errors="repladm@company.com, dbadmin@company.com"
```

**Example 5**

To start a Replication Alert Monitor that runs every 120 seconds (two minutes) and waits 1 440 minutes (24 hours) before sending alerts:

```
asnmon monitor_server=wsdb monitor_qual=monqual monitor_interval=120
      max_notifications_per_alert=2 max_notifications_minutes=1440
```

This Replication Alert Monitor program sends a maximum of two alerts when the alerts occurred during the time period specified by the `max_notifications_minutes` parameter value (1 440 minutes).

**Related tasks:**

- Chapter 11, “Monitoring replication” on page 161

**Related reference:**

- “asnmcmd: Operating the Replication Alert Monitor (UNIX, Windows, z/OS)” on page 329

---

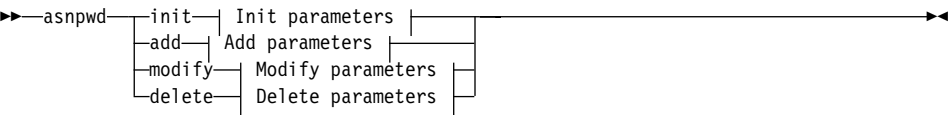
**asnpwd: Maintaining password files (UNIX and Windows)**

Use the **asnpwd** command to create and change password files on UNIX and Windows. Run this command at the command line or in a shell script.

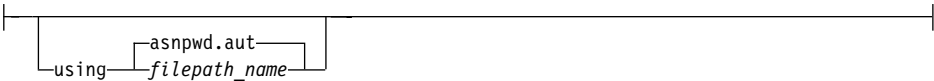
The parameter keywords of this command are *not* case sensitive.

Command help appears if you enter the **asnpwd** command without any parameters, followed by a `?`, or followed by incorrect parameters.

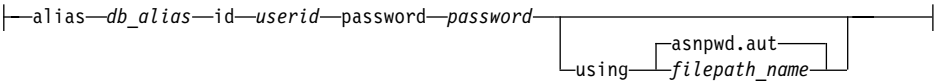
***To maintain password files using the asnpwd command:***



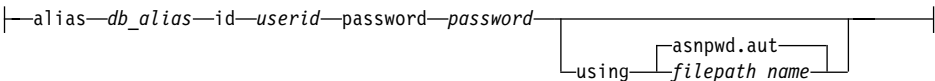
**Init parameters:**



**Add parameters:**



Modify parameters:



Delete parameters:

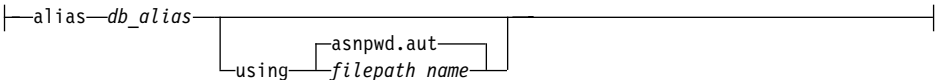


Table 26 defines the invocation parameters.

Table 26. asnpwd invocation parameter definitions for UNIX and Windows operating systems

Parameter	Definition
<b>init</b>	Specify to create an empty password file. This command will fail if you specify the <b>init</b> parameter with a password file that already exists.
<b>add</b>	Specify to add an entry to the password file. This command will fail if you specify the <b>add</b> parameter with an entry that already exists in the password file. Use the <b>modify</b> parameter to change an existing entry in the password file.
<b>modify</b>	Specify to modify the password or user ID for an entry in the password file.
<b>delete</b>	Specify to delete an entry from the password file.
<b>using filepath_name</b>	<p>Specifies the path and name of the password file. Follow the file naming conventions of your operating system. An example of a valid password file on Windows is C:\sqllib\mypwd.aut.</p> <p>If you specify the path and name of the password file, the path and the password file must already exist. If you are using the <b>init</b> parameter and you specify the path and name of the password file, the path must already exist and the command will create the password file for you.</p> <p>If you do not specify this parameter, the default file name is asnpwd.aut and the default file path is the current directory.</p>
<b>alias db_alias</b>	Specifies the alias of the database to which the user ID has access. The alias is always folded to uppercase, regardless of how it is entered.

Table 26. *asnpwd* invocation parameter definitions for UNIX and Windows operating systems (continued)

Parameter	Definition
<b>id</b> <i>userid</i>	Specifies the user ID that has access to the database.
<b>password</b> <i>password</i>	Specifies the password for the specified user ID. This password is case sensitive and is encrypted in the password file.

## Examples for asnpwd

The following examples illustrate how to use the **asnpwd** command.

### Example 1

To create a password file with the default name of `asnpwd.aut` in the current directory:

```
asnpwd INIT
```

### Example 2

To create a password file named `pass1.aut` in the `c:\myfiles` directory:

```
asnpwd INIT Using c:\myfiles\pass1.aut
```

### Example 3

To add a user ID called `oneuser` and its password to the password file named `pass1.aut` in the `c:\myfiles` directory and to grant this user ID access to the `db1` database:

```
asnpwd ADD ALIAS db1 ID oneuser PASSWORD mypwd using c:\myfiles\pass1.aut
```

### Example 4

To modify the user ID or password of an entry in the password file named `pass1.aut` in the `c:\myfiles` directory:

```
asnpwd MODIFY AliaS sample ID chglocalid PASSWORD chgmajorpwd  
USING c:\myfiles\pass1.aut
```

### Example 5

To delete the database alias called `sample` from the password file named `pass1.aut` in the `c:\myfiles` directory:

```
asnpwd delete aLiAs sample USING c:\myfiles\pass1.aut
```

### Example 6

To see command help:

```
asnpwd
```

### Related reference:

- “GRTDPRAUT: Authorizing users (OS/400)” on page 402

**asnscri: Creating a DB2 Replication Service to start Capture, Apply, or the Replication Alert Monitor (Windows only)**

Use the **asnscri** command to create a DB2 Replication Service in the Windows Service Control Manager (SCM) and invoke the **asncri**, **asnapply**, or **asnmon** command. Run the **asnscri** command on the Windows NT or the Windows 2000 operating system.

**To start a Capture, Apply, or Replication Alert Monitor program through a DB2 Replication Service using the asnscri command:**

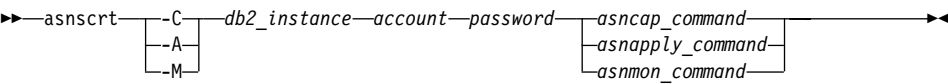


Table 27 defines the invocation parameters.

Table 27. *asnscri* invocation parameter definitions for Windows operating systems

Parameter	Definition
-C	Specifies that you are starting a Capture program.
-A	Specifies that you are starting an Apply program.
-M	Specifies that you are starting a Replication Alert Monitor program.
db2_instance	Specifies the DB2 instance used to identify a unique DB2 Replication Service. The DB2 instance can be a maximum of eight characters.
account	Specifies the account name that you use to log on to Windows. The account name must begin with a period and a backslash (.\\).
password	Specifies the password used with the account name. If the password contains special characters, type a backslash (\) before each special character.
asncri_command	Specifies the complete <b>asncri</b> command to start a Capture program. Use the documented <b>asncri</b> command syntax with the appropriate <b>asncri</b> parameters.  <b>Important:</b> If the DB2PATH environment variable is not defined, you must specify a location for the work files by including the <b>capture_path</b> parameter with the <b>asncri</b> command. If the DB2PATH variable is defined and you specify a <b>capture_path</b> , the <b>capture_path</b> parameter overrides the DB2PATH variable.  The <b>asnscri</b> command does not validate the syntax of the <b>asncri</b> parameters that you enter.



Table 27. *asnscri invocation parameter definitions for Windows operating systems (continued)*

Parameter	Definition
<i>asnapply_command</i>	<p>Specifies the complete <b>asnapply</b> command to start an Apply program. Use the documented <b>asnapply</b> command syntax with the appropriate <b>asnapply</b> parameters.</p> <p><b>Important:</b> If the DB2PATH environment variable is not defined, you must specify the location for the work files by including the <b>apply_path</b> parameter with the <b>asnapply</b> command. If the DB2PATH variable is defined and you specify an <b>apply_path</b>, the <b>apply_path</b> parameter overrides the DB2PATH variable.</p> <p>The <b>asnscri</b> command does not validate the syntax of the <b>asnapply</b> parameters that you enter.</p>
<i>asnmon_command</i>	<p>Specifies the complete <b>asnmon</b> command to start a Replication Alert Monitor program. Use the documented <b>asnmon</b> command syntax with the appropriate <b>asnmon</b> parameters.</p> <p><b>Important:</b> If the DB2PATH environment variable is not defined, you must specify a location for the log files by including the <b>monitor_path</b> parameter with the <b>asnmon</b> command. If the DB2PATH variable is defined and you specify a <b>monitor_path</b>, the <b>monitor_path</b> parameter overrides the DB2PATH variable.</p> <p>The <b>asnscri</b> command does not validate the syntax of the <b>asnmon</b> parameters that you enter.</p>

## Examples for asnscri

The following examples illustrate how to use the **asnscri** command.

### Example 1

To create a DB2 Replication Service that invokes a Capture program under a DB2 instance called inst1:

```
asnscri -C inst1 .\joesmith password asncap capture_server=sampled
capture_schema=ASN capture_path=X:\logfiles
```

### Example 2

To create a DB2 Replication Service that invokes an Apply program under a DB2 instance called inst2 using a logon account of .\joesmith and a password of my\$pwd:

```
asnscri -A inst2 .\joesmith my\my$pwd asnapply control_server=db2 apply_qual=aq2
apply_path=X:\sqllib
```

**Example 3**

To create a DB2 Replication Service that invokes a Replication Alert Monitor program under a DB2 instance called inst3:

```
asnscri -M inst3 .\joesmith password asnmon monitor_server=db3 monitor_qual=mq3
monitor_path=X:\logfiles
```

**Example 4**

To create a DB2 Replication Service that invokes a Capture program under a DB2 instance called inst4 and overrides the default work file directory with a fully qualified **capture\_path**:

```
asnscri -C inst4 .\joesmith password X:\sqllib\bin\asnscap capture_server=scdb
capture_schema=ASN capture_path=X:\logfiles
```

**Related tasks:**

- Chapter 20, “Using the Windows Service Control Manager to issue system commands (Windows)” on page 459

**Related reference:**

- “asnsdrop: Dropping a DB2 Replication Service (Windows only)” on page 340
- “asnapply: Starting Apply (UNIX, Windows, z/OS)” on page 308
- “asnscap: Starting Capture (UNIX, Windows, z/OS)” on page 316
- “asnmon: Starting the Replication Alert Monitor (UNIX, Windows, z/OS)” on page 330

---

**asnsdrop: Dropping a DB2 Replication Service (Windows only)**

Use the **asnsdrop** command to drop a DB2 Replication Service from the Windows Service Control Manager (SCM) on the Windows NT or the Windows 2000 operating system. (You create a DB2 Replication Service using the **asnscri** command.)

*To drop a DB2 Replication Service using the asnsdrop command:*

```
►►—asnsdrop—service_name—◄◄
```

Table 28 defines the invocation parameters.

*Table 28. asnsdrop invocation parameter definitions for Windows operating systems*

Parameter	Definition
<i>service_name</i>	Specifies the fully qualified name of the DB2 Replication Service. Enter the Windows SCM to obtain the DB2 Replication Service name. On Windows 2000 operating systems, you can obtain the service name by opening the Properties window of the DB2 Replication Service.  If the DB2 Replication Service name contains spaces, enclose the entire service name in double quotation marks.

## Examples for asnsdrop

The following examples illustrate how to use the **asnsdrop** command.

### Example 1

To drop a DB2 Replication Service:

```
asnsdrop DB2.SAMPLEDB.SAMPLEDB.CAP.ASN
```

### Example 2

To drop a DB2 Replication Service with a schema named A S N:

```
asnsdrop "DB2.SAMPLEDB.SAMPLEDB.CAP.A S N"
```

### Related tasks:

- Chapter 20, “Using the Windows Service Control Manager to issue system commands (Windows)” on page 459

### Related reference:

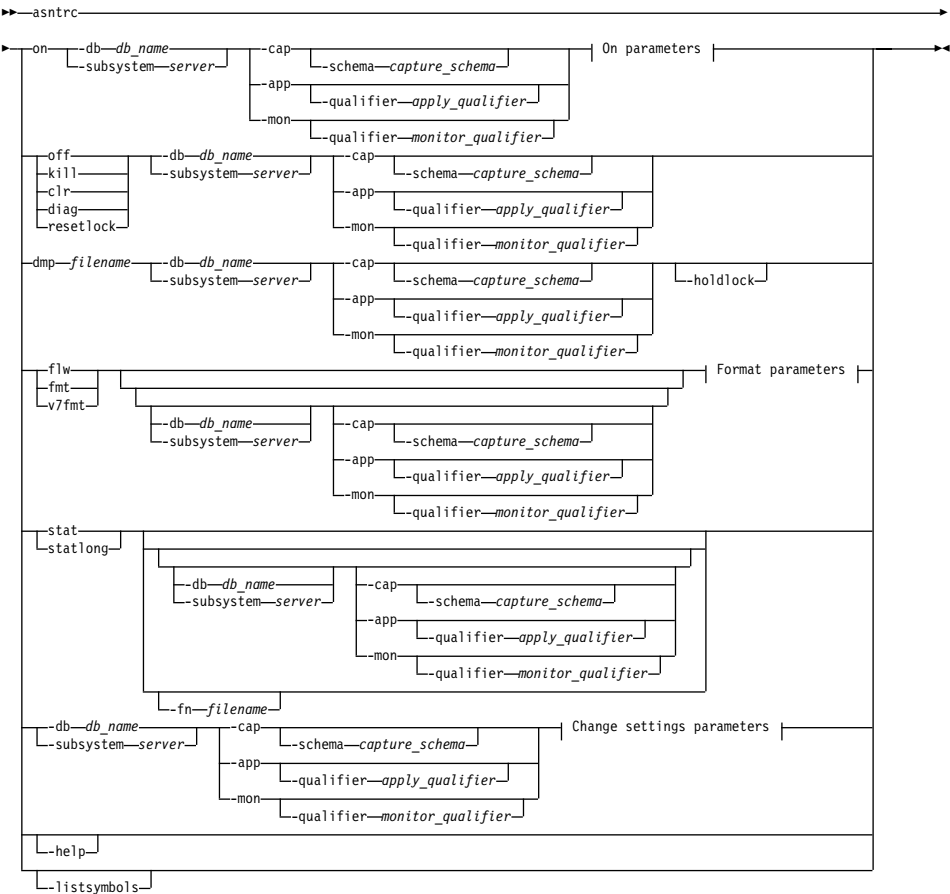
- “asnsrct: Creating a DB2 Replication Service to start Capture, Apply, or the Replication Alert Monitor (Windows only)” on page 338

---

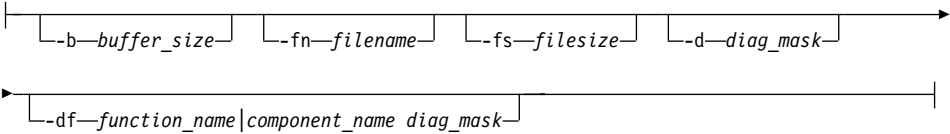
## asntrc: Operating the replication trace facility (UNIX, Windows, z/OS)

Use the **asntrc** command to run the trace facility on UNIX, Windows, and UNIX System Services (USS) on z/OS. The trace facility logs program flow information from Capture, Apply, and Replication Alert Monitor programs. You can provide this trace information to IBM Software Support for troubleshooting assistance. Run this command at an operating system prompt or in a shell script.

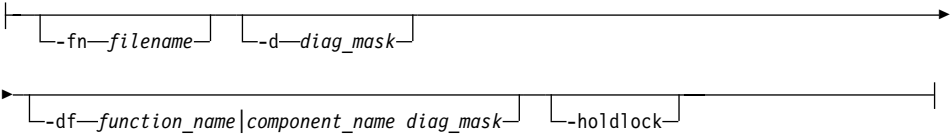
***To run the trace facility using the asntrc command:***



On parameters:



Format parameters:



## Change settings parameters:

```
|-----|
|_d_diag_mask|_df_function_name|component_name diag_mask|
|-----|
```

Table 29 defines the invocation parameters.

Table 29. *asntrc* invocation parameter definitions for UNIX, Windows, and z/OS operating systems

Parameter	Definition
<b>on</b>	Specify to turn on the trace facility for a specific Capture, Apply, or Replication Alert Monitor program. The trace facility creates a shared memory segment used during the tracing process.
<b>-db db_name</b>	<b>For UNIX and Windows only:</b> Specifies the database name under which the Apply or Capture program to be traced resides.
<b>-subsystem server</b>	<b>For z/OS only:</b> Specifies the name of the database server under which the Capture, Apply, or Replication Alert Monitor program to be traced resides.
<b>-cap</b>	Specifies that a Capture program is to be traced. The Capture program is identified by the <b>-schema</b> parameter.
<b>-schema capture_schema</b>	Specifies the name of the Capture program to be traced. The Capture program is identified by the Capture schema that you enter. Use this parameter with the <b>-cap</b> parameter.
<b>-app</b>	Specifies that an Apply program is to be traced. The Apply program is identified by the <b>-qualifier</b> parameter.
<b>-qualifier apply_qualifier</b>	Specifies the name of Apply program to be traced. This Apply program is identified by the Apply qualifier that you enter. Use this parameter with the <b>-app</b> parameter.
<b>-mon</b>	Specifies that a Replication Alert Monitor program is to be traced. The Replication Alert Monitor program is identified by the <b>-qualifier</b> parameter.
<b>-qualifier monitor_qualifier</b>	Specifies the name of Replication Alert Monitor program to be traced. This Replication Alert Monitor program is identified by the monitor qualifier that you enter. Use this parameter with the <b>-mon</b> parameter.

Table 29. *asntrc* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>off</b>	Specify to turn off the trace facility for a specific Capture, Apply, or Replication Alert Monitor program and free the shared memory segment in use.
<b>kill</b>	Specify to force an abnormal termination of the trace facility.  Use this parameter only if you encounter a problem and are unable to turn the trace facility off with the <b>off</b> parameter.
<b>clr</b>	Specify to clear a trace buffer. This parameter erases the contents of the trace buffer but leaves the buffer active.
<b>diag</b>	Specify to view the filter settings while the trace facility is running.
<b>resetlock</b>	Specify to release the buffer latch of a trace facility. This parameter enables the buffer latch to recover from an error condition in which the trace program terminated while holding the buffer latch.
<b>dmp filename</b>	Specify to write the current contents of the trace buffer to a file.
<b>-holdlock</b>	Specifies that the trace facility can complete a file dump or output command while holding a lock, even if the trace facility finds insufficient memory to copy the buffer.
<b>flw</b>	Specify to display summary information produced by the trace facility and stored in shared memory or in a file. This information includes the program flow and is displayed with indentations that show the function and call stack structures for each process and thread.
<b>fmt</b>	Specify to display detailed information produced by the trace facility and stored in shared memory or in a file. This parameter displays the entire contents of the traced data structures in chronological order.
<b>v7fmt</b>	Specify to display information produced by the trace facility and stored in shared memory or in a file. This trace information appears in Version 7 format.

Table 29. *asntrc* invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition
<b>stat</b>	Specify to display the status of a trace facility. This status information includes the trace version, application version, number of entries, buffer size, amount of buffer used, status code, and program timestamp.
<b>statlong</b>	Specify to display the status of a trace facility with additional z/OS version level information. This additional information includes the service levels of each module in the application and appears as long strings of text.
<b>-fn filename</b>	Specifies the file name containing the mirrored trace information, which includes all the output from the trace facility.
<b>-help</b>	Displays the valid command parameters with descriptions.
<b>-listsymbols</b>	Displays the valid function and component identifiers to use with the <b>-df</b> parameter.
<b>-b buffer_size</b>	Specifies the size of the trace buffer (in bytes). You can enter a K or an M after the number to indicate kilobytes or megabytes, respectively; these letters are not case sensitive.
<b>-fs filesize</b>	Specifies the size limit (in bytes) of the mirrored trace information file.

Table 29. asntrc invocation parameter definitions for UNIX, Windows, and z/OS operating systems (continued)

Parameter	Definition								
<b>-d</b> <i>diag_mask</i>	<p>Specifies the types of trace records to be recorded by the trace facility. Trace records are categorized by a diagnostic mask number:</p> <table><tr><td>1</td><td>Flow data, which includes the entry and exit points of functions.</td></tr><tr><td>2</td><td>Basic data, which includes all major events encountered by the trace facility.</td></tr><tr><td>3</td><td>Detailed data, which includes the major events with descriptions.</td></tr><tr><td>4</td><td>Performance data.</td></tr></table> <p><b>Important:</b> The higher diagnostic mask numbers are <i>not</i> inclusive of the lower diagnostic mask numbers.</p> <p>You can enter one or more of these numbers to construct a diagnostic mask that includes only the trace records that you need. For example, specify <b>-d 4</b> to record only performance data; specify <b>-d 1,4</b> to record only flow and performance data; specify <b>-d 1,2,3,4</b> (the default) to record all trace records. Separate the numbers with commas.</p> <p>Enter a diagnostic mask number of 0 (zero) to specify that no global trace records are to be recorded by the trace facility. Type <b>-d 0</b> to reset the diagnostic level before specifying new diagnostic mask numbers for a tracing facility.</p>	1	Flow data, which includes the entry and exit points of functions.	2	Basic data, which includes all major events encountered by the trace facility.	3	Detailed data, which includes the major events with descriptions.	4	Performance data.
1	Flow data, which includes the entry and exit points of functions.								
2	Basic data, which includes all major events encountered by the trace facility.								
3	Detailed data, which includes the major events with descriptions.								
4	Performance data.								
<b>-df</b> <i>function_name / component_name</i> <i>diag_mask</i>	<p>Specifies that a particular function or component identifier is to be traced.</p> <p>Type the diagnostic mask number (1,2,3,4) after the function or component identifier name. You can enter one or more of these numbers. Separate the numbers with commas.</p>								

Examples for asntrc

The following examples illustrate how to use the **asntrc** command.

Example 1

To trace a running Capture program under UNIX or Windows:

1. Start the trace facility, specifying a trace file name with a maximum buffer and file size:



```
asntrc on -db mydb -cap -schema myschema -b 256k -fn myfile.trc -fs 500m
```

2. Start the Capture program, and let it run for an appropriate length of time.
3. While the trace facility is on, display the data directly from shared memory.

To display the summary process and thread information from the trace facility:

```
asntrc flw -db mydb -cap -schema myschema
```

To view the flow, basic, detailed, and performance data records only from the Capture log reader:

```
asntrc fmt -db mydb -cap -schema myschema -d 0
-df "Capture Log Read" 1,2,3,4
```

4. Stop the trace facility:

```
asntrc off -db mydb -cap -schema myschema
```

The trace file contains all of the Capture program trace data that was generated from the start of the Capture program until the trace facility was turned off.

5. After you stop the trace facility, format the data from the generated binary file:

```
asntrc flw -fn myfile.trc
```

and

```
asntrc fmt -fn myfile.trc -d 0 -df "Capture Log Read" 1,2,3,4
```

### Example 2

To start a trace facility of a Replication Alert Monitor program running under UNIX or Windows:

```
asntrc on -db mydb -mon -qualifier monq
```

### Example 3

To trace only performance data of an Apply program running under UNIX or Windows:

```
asntrc on -db mydb -app -qualifier aql -b 256k -fn myfile.trc -d 4
```

### Example 4

To trace all flow and performance data of a Capture program running under z/OS:

```
asntrc on -subsystem dbserve1 -cap -schema myschema -b 256k
-fn myfile.trc -d 1,4
```

### Example 5

To trace all global performance data and the specific Capture log reader flow data of a Capture program running under UNIX or Windows:

## asntrc

```
asntrc on -db mydb -cap -schema myschema -b 256k -fn myfile.trc -d 4  
-df "Capture Log Read" 1
```

### Example 6

To trace a running Capture program under UNIX or Windows and then display and save a point-in-time image of the trace facility:

1. Start the trace command, specifying a buffer size large enough to hold the latest records:

```
asntrc on -db mydb -cap -schema myschema -b 4m
```

2. Start the Capture program, and let it run for an appropriate length of time.
3. View the detailed point-in-time trace information that is stored in shared memory:

```
asntrc fmt -db mydb -cap -schema myschema
```

4. Save the point-in-time trace information to a file:

```
asntrc dmp myfile.trc -db mydb -cap -schema myschema
```

5. Stop the trace facility:

```
asntrc off -db mydb -cap -schema myschema
```

### Related reference:

- “WRKDPTRTC: Using the DPR trace facility (OS/400)” on page 446

---

## Chapter 18. System commands for replication (OS/400)

This chapter describes the replication commands that run under the OS/400 operating system on iSeries servers. You can enter these commands at an operating system command prompt or through a command line program.

This chapter contains a section for each command. Each section contains a brief description of the command, a syntax diagram, and a table of parameters with corresponding definitions. The end of each section has examples of command usage and cross-references to related information.

The commands include:

- “ADDDPRREG: Adding a DPR registration (OS/400)”
- “ADDDPRSUB: Adding a DPR subscription set (OS/400)” on page 359
- “ADDDPRSUBM: Adding a DPR subscription-set member (OS/400)” on page 376
- “ANZDPR: Operating the Analyzer (OS/400)” on page 387
- “CHGDPRCAPA: Changing DPR Capture attributes (OS/400)” on page 391
- “CRTDPRTBL: Creating the replication control tables (OS/400)” on page 396
- “ENDDPRAPY: Stopping Apply (OS/400)” on page 397
- “ENDDPRCAP: Stopping Capture (OS/400)” on page 400
- “GRTPRAUT: Authorizing users (OS/400)” on page 402
- “INZDPRCAP: Reinitializing DPR Capture (OS/400)” on page 412
- “OVRDPRCAPA: Overriding DPR capture attributes (OS/400)” on page 414
- “RMVDPRREG: Removing a DPR registration (OS/400)” on page 420
- “RMVDPRSUB: Removing a DPR subscription set (OS/400)” on page 421
- “RMVDPRSUBM: Removing a DPR subscription-set member (OS/400)” on page 423
- “RVKDPRAUT: Revoking authority (OS/400)” on page 425
- “STRDPRAPY: Starting Apply (OS/400)” on page 427
- “STRDPRCAP: Starting Capture (OS/400)” on page 436
- “WRKDPTRC: Using the DPR trace facility (OS/400)” on page 446

---

### **ADDDPRREG: Adding a DPR registration (OS/400)**

Use the Add DPR registration (**ADDDPRREG**) command to register a table as a source table for DB2 DataPropagator for iSeries.

## ADDDPRREG

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

### **To register a table using the ADDDPRREG command:**

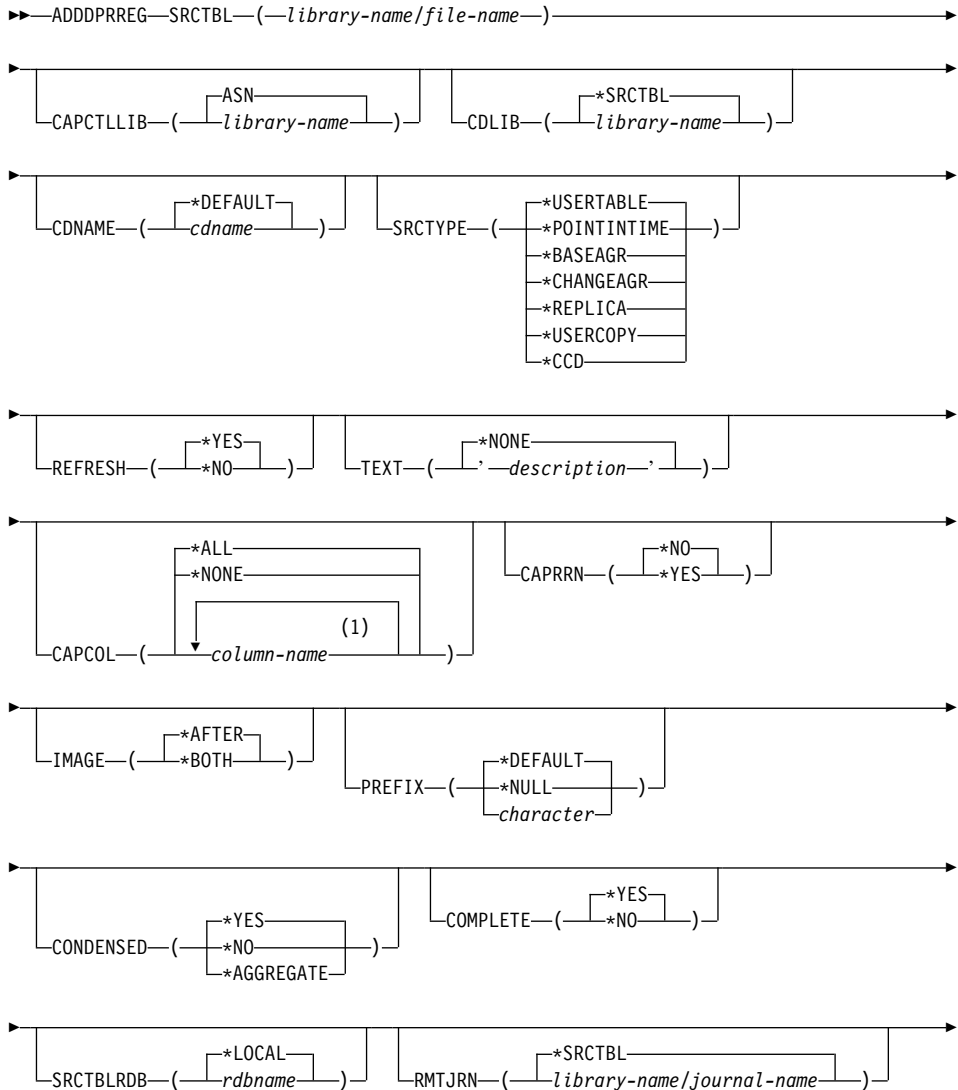




Table 30. ADDDPRREG command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
CDNAME	<p>Specifies the name of the change-data (CD) table.</p> <p><b>*DEFAULT</b> (default)</p> <p>Creates the CD table with the default name, which is based on the current timestamp. For example, if the current timestamp is January 23, 2002 at 09:58:26, the default name is ASN020123095826CD.</p> <p><i>cdname</i></p> <p>Creates the CD table with this specified name.</p>
SRCTYPE	<p>Specifies the type of source table that you are registering. Choose a source type based on your replication configuration:</p> <ul style="list-style-type: none"><li>• Use the default of USERTABLE for a basic data distribution or a data consolidation configuration.</li><li>• Use REPLICA for an update-anywhere configuration.</li><li>• Use POINTINTIME, BASEAGR, CHANGEAGR, USERCOPY, or CCD if you have a multi-tier configuration and want the target table to be a source for a subsequent tier in your replication configuration.</li></ul> <p>If you are registering an existing target table as a source, the registration fails if the target table does not contain the IBMSNAP table columns indicated by the specified source type.</p>

Table 30. ADDDPRREG command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>SRCTYPE</b> (continued)	<p><b>*USERTABLE</b> (default) A user database table, which is the most common type of registered table. The table cannot contain any columns that start with a DB2 DataPropagator for iSeries column identifier of either IBMSNAP or IBMQSQ.</p> <p><b>*POINTINTIME</b> A point-in-time copy table, which includes content that matches all or part of the content of a source table and a DB2 DataPropagator for iSeries system column that identifies the time when a particular row was last inserted or updated at the source system. The table must contain the IBMSNAP_LOGMARKER timestamp column and can optionally contain an INTEGER column called IBMQSQ_RRN.</p> <p><b>*BASEAGR</b> A base aggregate copy, which contains data aggregated at intervals from a user table or from a point-in-time table. The base aggregate table must contain the IBMSNAP_HLOGMARKER and IBMSNAP_LLOGMARKER timestamp columns.</p> <p><b>*CHANGEAGR</b> A change aggregate copy table, which contains data aggregations that are based on changes recorded for a source table. The table must contain the IBMSNAP_HLOGMARKER and IBMSNAP_LLOGMARKER timestamp columns.</p> <p><b>*REPLICA</b> A target table for a replica subscription. Register this type of table so that changes from the target table can be fed back to the original source table. This table cannot contain any DB2 DataPropagator for iSeries system columns or any columns that start with the DB2 DataPropagator for iSeries column identifier of either IBMSNAP or IBMQSQ. The table contains all of the columns from the original source table.</p> <p><b>*USERCOPY</b> A target table with content that matches all or part of the content of a source table. The user copy table contains only user data columns.</p>

Table 30. ADDDPRREG command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>SRCTYPE</b> (continued)	<p><b>*CCD</b></p> <p>A consistent-change data (CCD) table, which contains transaction-consistent data from the source table. The table must contain columns that are defined as follows:</p> <ul style="list-style-type: none"><li>• IBMSNAP_INTENTSEQ CHAR(10) FOR BIT DATA NOT NULL</li><li>• IBMSNAP_OPERATION CHAR(1) NOT NULL</li><li>• IBMSNAP_COMMITSEQ CHAR(10) FOR BIT DATA NOT NULL</li><li>• IBMSNAP_LOGMARKER TIMESTAMP NOT NULL</li></ul>
<b>REFRESH</b>	<p>Specifies whether the full-refresh capability is enabled. You can use this value to turn off the capability of the Apply program to perform a full refresh from the source database.</p> <p><b>*YES</b> (default) Full refreshes are allowed.</p> <p><b>*NO</b> Full refreshes are not allowed.</p> <p>If the target table is a base aggregate or change aggregate, you should set this parameter to *NO.</p>
<b>TEXT</b>	<p>Specifies the textual description that is associated with this registration.</p> <p><b>*NONE</b> (default) No description is associated with the entry.</p> <p><i>description</i> The textual description of this registration. You can enter a maximum of 50 characters and must enclose the text in single quotation marks.</p>



Table 30. ADDDPRREG command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>CAPCOL</b>	<p>Specifies the columns for which changes are captured for this registered table.</p> <p><b>*ALL</b> (default) Changes are captured for all columns.</p> <p><b>*NONE</b> Changes are not captured for this table. Use this value to specify that you want this table registered for full refresh only. The change-data (CD) table is not created with this registered table, and the Capture program will not capture changes for the table.</p> <p><i>column-name</i> The column names for which changes are captured. You can type up to 300 column names. Separate the column names with spaces.</p>
<b>CAPRRN</b>	<p>Specifies whether the relative record number (RRN) of each changed record is captured.</p> <p><b>*NO</b> (default) The relative record number is not captured.</p> <p><b>*YES</b> The relative record number is captured. An additional column called IBMQSQ_RRN is created in the change-data (CD) table.  Set this parameter to *YES only if there are no unique keys in the source table.</p>
<b>IMAGE</b>	<p>Specifies whether the change-data (CD) table contains both before and after images of the changes to the source table. This applies globally to all columns specified on the Capture columns (<b>CAPCOL</b>) parameter.</p> <p>This <b>IMAGE</b> parameter is not valid when the <b>CAPCOL</b> parameter is set to *NONE.</p> <p>The source table must be journaled with *BOTH images even if you specify *AFTER on this parameter.</p> <p><b>*AFTER</b> (default) The Capture program records only after images of the source table in the CD table.</p> <p><b>*BOTH</b> The Capture program records both before images and after images of the source table in the CD table.</p>

Table 30. ADDDPRREG command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>PREFIX</b>	<p>Specifies the prefix character identifying before-image column names in the change-data (CD) table. You must ensure that none of the registered column names of the source table begins with this prefix character.</p> <p><b>*DEFAULT</b> (default) The default prefix (@) is used.</p> <p><b>*NULL</b> No before images are captured. This value is not valid if the <b>IMAGE</b> parameter is set to <b>*BOTH</b>.</p> <p><i>character</i> Any single alphabetic character that is valid in an object name.</p>
<b>CONDENSED</b>	<p>Specifies whether the source table is condensed. A condensed table contains current data with no more than one row for each primary key value in the table.</p> <p><b>*YES</b> (default) The source table is condensed.</p> <p><b>*NO</b> The source table is not condensed.</p> <p><b>*AGGREGATE</b> The source table type is either <b>*BASEAGR</b> (base aggregate) or <b>*CHANGEAGR</b> (change aggregate). If this value is used, you must set the <b>COMPLETE</b> parameter to <b>*NO</b>.</p>
<b>COMPLETE</b>	<p>Specifies whether the source table is complete, which means that the table contains a row for every primary key value of interest.</p> <p><b>*YES</b> (default) The source table is complete.</p> <p><b>*NO</b> The source table is not complete.</p>
<b>SRCTBLRDB</b>	<p>Specifies whether you want to use remote journaling, in which the source table and the remote journal reside on different systems. Use this parameter to specify the location of the source table.</p> <p><b>*LOCAL</b> (default) The source table resides locally (on the machine where you are running the <b>ADDDPRREG</b> command).</p> <p><i>rdbname</i> The name of the relational database where the source table exists. You can use the Work with RDB Directory Entries (<b>WRKRDBDIRE</b>) command to find this relational database name.</p>

Table 30. ADDDPRREG command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>RMTJRN</b>	<p>Specifies the name of the remote journal when the name of this journal and the name of the journal on the source system are different. You must issue this command from the system where the remote journal resides.</p> <p><b>*SRCTBL</b> (default) The remote journal name is the same as the journal name of the source table.</p> <p><i>library-name/journal-name</i> The qualified library and journal name that reside on this system and are used for journaling the remote source table.</p> <p>You can specify a remote journal name only if you specified a remote source table location using the <b>SRCTBLRDB</b> parameter.</p>
<b>CONFLICT</b>	<p>Specifies the conflict level that is used by the Apply program when detecting conflicts in a replica subscription.</p> <p><b>*NONE</b> (default) No conflict detection.</p> <p><b>*STANDARD</b> Moderate conflict detection. The Apply program searches for conflicts in rows that are already captured in the replica change-data (CD) tables.</p> <p><b>*ENHANCED</b> Enhanced conflict detection. This option provides the best data integrity among all replicas and source tables.</p>
<b>UPDDELINS</b>	<p>Determines how the Capture program stores updated source data in the change-data (CD) table.</p> <p><b>*NO</b> (default) The Capture program stores each source change in a single row in the CD table.</p> <p><b>*YES</b> The Capture program stores each source change using two rows in the CD table, one for the delete and one for the insert. The Apply program processes the delete row first and the insert row second.</p>

Table 30. ADDDPRREG command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>GENCDROW</b>	<p>Specifies whether the Capture program captures changes from all rows in the source table.</p> <p><b>*ALLCHG</b> (default) The Capture program captures changes from all rows in the source table (including changes in unregistered columns) and adds these changes to the change-data (CD) table.</p> <p><b>*REGCOLCHG</b> The Capture program captures changes only if the changes occur in registered columns. The Capture program then adds these rows to the CD table.</p> <p>You cannot specify *REGCOLCHG if the <b>CAPCOL</b> parameter is set to *ALL or *NONE.</p>
<b>RECAP</b>	<p>Specifies whether the changes made by the Apply program are recaptured by the Capture program.</p> <p><b>*YES</b> (default) Changes made to the source table by the Apply program are captured and entered into the change-data (CD) table.</p> <p><b>*NO</b> Changes that were made to the source table by the Apply program are not captured and, therefore, do not appear in the CD table. You should use this option when registering REPLICA type tables.</p>
<b>STOPONERR</b>	<p>Specifies whether the Capture program stops when it encounters an error.</p> <p><b>*NO</b> (default) The Capture program does not stop when it encounter an error. The Capture program issues messages, deactivates the registration that caused the error, and then continues processing.</p> <p><b>*YES</b> The Capture program issues messages and then stops when it encounters an error.</p>

Examples for ADDDPRREG

The following examples illustrate how to use the **ADDDPRREG** command.

Example 1

To register a source table named **EMPLOYEE** from the **HR** library under the default Capture schema:

```
ADDDPRREG SRCTBL(HR/EMPLOYEE)
```

### Example 2

To register a source table named EMPLOYEE from the HR library under the BSN Capture schema and to create a CD table named CDEMPLOYEE under the HRCDLIB library:

```
ADDDPRREG SRCTBL(HR/EMPLOYEE) CAPCTLLIB(BSN) CDLIB(HRCDLIB) CDNAME(CDEMPLOYEE)
```

### Example 3

To register a point-in-time copy source table named SALES from the DEPT library under the BSN Capture schema:

```
ADDDPRREG SRCTBL(DEPT/SALES) CAPCTLLIB(BSN) SRCTYPE(*POINTINTIME)
```

### Example 4

To register a source table named SALES from the DEPT library and to specify that the CD table contains both before and after images of source table changes:

```
ADDDPRREG SRCTBL(DEPT/SALES) IMAGE(*BOTH)
```

### Example 5

To register a source table named SALES from the DEPT library of the relational database named RMTRDB1 using remote journals:

```
ADDDPRREG SRCTBL(DEPT/SALES) SRCTBLRDB(RMTRDB1) RMTJRN(RMTJRNLIB/RMTJRN)
```

### Example 6

To register the EMPLOYEE source table from the HR library and to capture changes only for the EMPNO, NAME, DEPT, and NETPAY columns:

```
ADDDPRREG SRCTBL(HR/EMPLOYEE) CAPCOL(EMPNO NAME DEPT NETPAY)
```

### Related tasks:

- Chapter 3, “Registering tables and views as replication sources” on page 37

---

## ADDDPRSUB: Adding a DPR subscription set (OS/400)

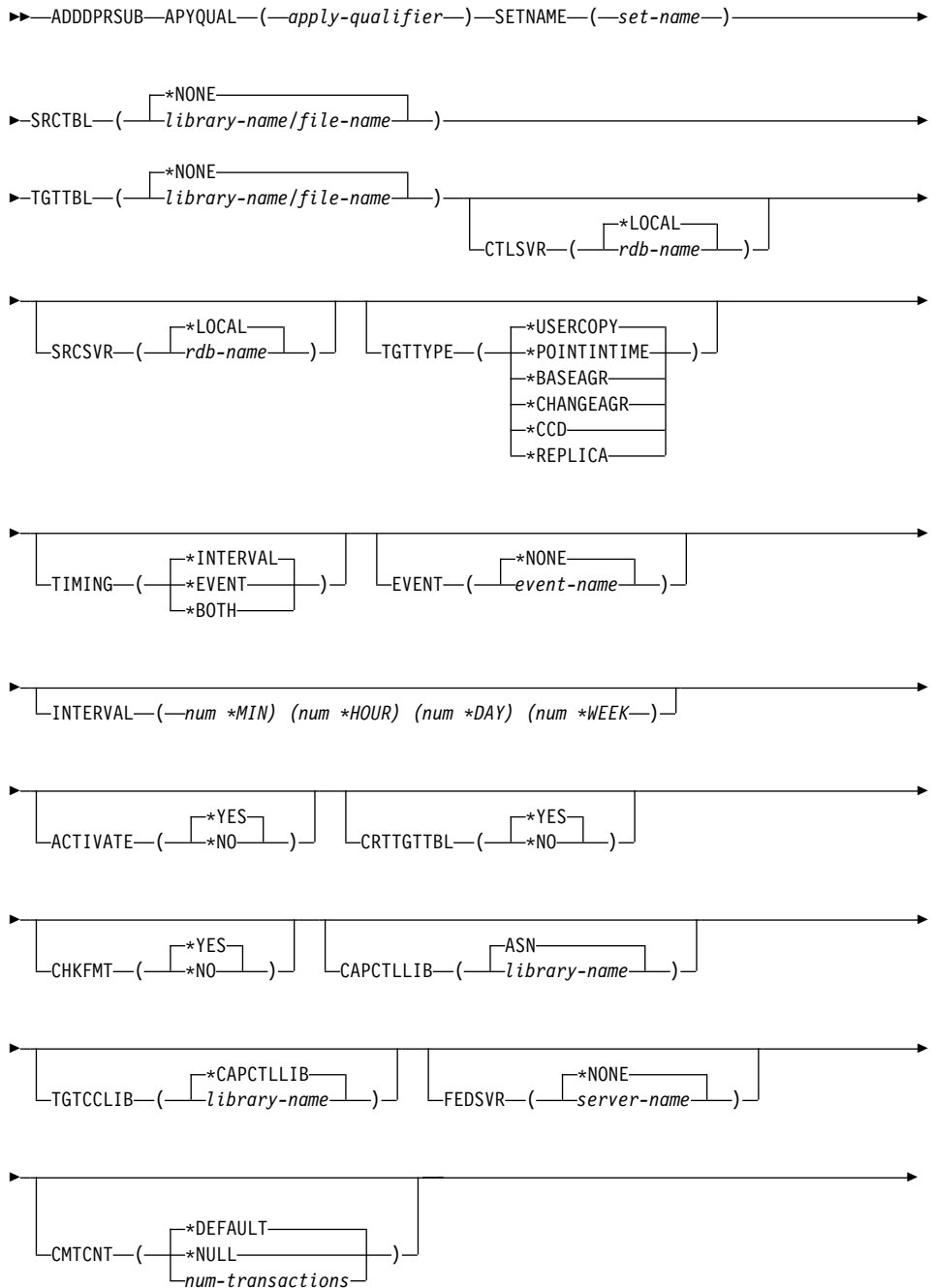
Use the Add DPR subscription set (**ADDDPRSUB**) command to create a subscription set with either one member or no members.

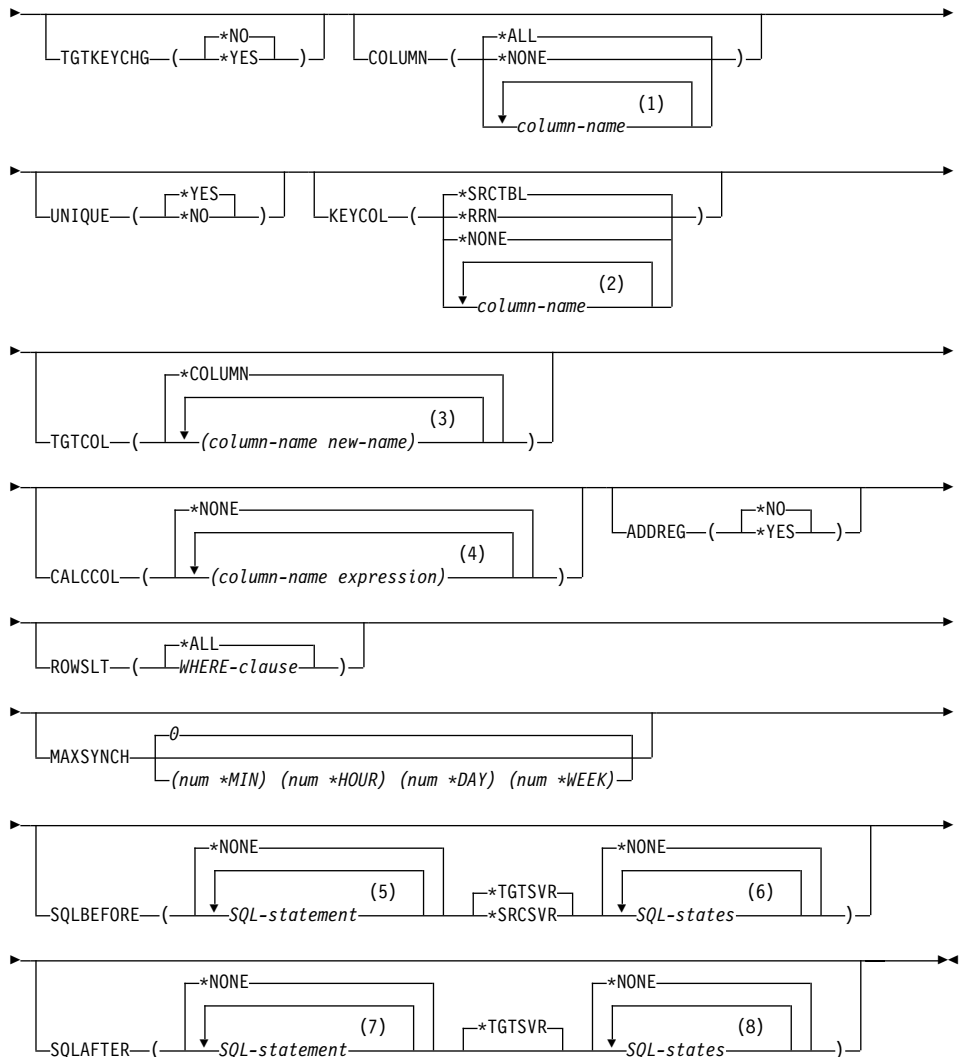
After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

***To create a subscription set using the ADDDPRSUB command:***

## ADDDPRSUB





### Notes:

- 1 You can specify up to 300 column names.
- 2 You can specify up to 120 column names.
- 3 You can specify up to 300 column names.
- 4 You can specify up to 100 column names and expressions.
- 5 You can specify up to 3 SQL statements.
- 6 You can specify up to 10 SQLSTATES.
- 7 You can specify up to 3 SQL statements.

8     You can specify up to 10 SQLSTATES.

Table 31. ADDDPRSUB command parameter definitions for OS/400

Parameter	Definition and prompts
APYQUAL	<p>Specifies the Apply qualifier that identifies which Apply program processes this subscription set. Subscription sets under an Apply qualifier run in a separate job. This parameter is required.</p> <p><i>apply-qualifier</i> The name of the Apply qualifier.</p>
SETNAME	<p>Specifies the subscription-set name. This parameter is required.</p> <p><i>set-name</i> The name of the subscription set. The subscription-set name that you enter must be unique for the specified Apply qualifier or the ADDDPRSUB command produces an error. Because the Apply program handles the set of target tables as a group, when one target table fails for any reason, the entire subscription set fails.</p>
SRCTBL	<p>Specifies the name of the source table that is used to copy information into your subscription set. You must register this table at the Capture control server before this table can become a member of a subscription set. This parameter is required.</p> <p>*NONE (default) This subscription set does not have a source member. Use when creating a subscription set without members.</p> <p><i>library-name/file-name</i> The qualified name of the source table. Use when creating a subscription set with one member.</p>
TGTTBL	<p>Specifies the name of the target table. The target table is automatically created if you set the CRITGTTBL parameter to *YES and the target table does not already exist. This parameter is required.</p> <p>*NONE (default) This subscription set does not have a target member. Use when creating a subscription set without members.</p> <p><i>library-name/file-name</i> The qualified name of the target table. Use when creating a subscription set with one member.</p>



Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>CTLSVR</b>	<p>Specifies the relational database name of the system that contains the Apply control tables.</p> <p><b>*LOCAL</b> (default) The Apply control tables reside locally (on the machine from which you are running the <b>ADDDPRSUB</b> command).</p> <p><i>rdb-name</i> The name of the relational database where the Apply control tables reside. You can use the Work with RDB Directory Entries (<b>WRKRDBDIRE</b>) command to find this name.</p>
<b>SRCSVR</b>	<p>Specifies the relational database name of the system that contains the Capture control tables.</p> <p><b>*LOCAL</b> (default) The source table is registered on the local machine (the machine from which you are running the <b>ADDDPRSUB</b> command).</p> <p><i>rdb-name</i> The name of the relational database where the Capture control tables reside. You can use the Work with RDB Directory Entries (<b>WRKRDBDIRE</b>) command to find this name.</p>
<b>TGTTYPE</b>	<p>Specifies the target table type. After you create a target table as one of these types, you can use this parameter value on the <b>SRCTBL</b> parameter of the Add DPR Registration (<b>ADDDPRREG</b>) command to register this target table as a source table for multi-tier replication.</p> <p><b>*USERCOPY</b> (default) The target table is a user copy, which is a target table with content that matches all or part of the content of a source table. A user copy is handled like a point-in-time copy but does not contain any of the DB2 DataPropagator for iSeries system columns that are present in the point-in-time target table.</p> <p>This value is not valid when a value of <b>*RRN</b> is specified on the <b>KEYCOL</b> parameter.</p> <p>The table that you specified with the <b>SRCTBL</b> parameter must be one of the following types: user database, point-in-time copy, or consistent-change data (CCD).</p> <p><b>Important:</b> If the target table already exists, DB2 DataPropagator for iSeries does not automatically journal changes to it. You must start journaling outside of DB2 DataPropagator for iSeries.</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
TGTTYPE (continued)	<p><b>*POINTINTIME</b></p> <p>The target table is a point-in-time copy. A point-in-time copy is a target table with content that matches all or part of the content of the source table and includes the DB2 DataPropagator for iSeries system column (IBMSNAP_LOGMARKER), which identifies when a particular row was inserted or updated at the Capture control server.</p> <p><b>*BASEAGR</b></p> <p>The target table is a base aggregate copy, which is a target table that contains data that is aggregated (calculated) from a source table. The source table for a base aggregate target must be either a user table or a point-in-time table. This target table must contain the IBMSNAP_HLOGMARKER and IBMSNAP_LLOGMARKER system timestamp columns.</p> <p><b>*CHANGEAGR</b></p> <p>The table is a change aggregate copy, which is a target table that contains data that is aggregated (calculated) based on the contents of a change-data (CD) table. This target table is created with the IBMSNAP_HLOGMARKER and IBMSNAP_LLOGMARKER system timestamp columns.</p> <p><b>*CCD</b></p> <p>The table is a consistent-change data (CCD) table, which is a target table created from a join of data in the change-data (CD) table and the unit-of-work (UOW) table. A CCD table provides transaction-consistent data for the Apply program and must include the following columns:</p> <ul style="list-style-type: none"><li>• IBMSNAP_INTENTSEQ</li><li>• IBMSNAP_OPERATION</li><li>• IBMSNAP_COMMITSEQ</li><li>• IBMSNAP_LOGMARKER</li></ul> <p><b>*REPLICA</b></p> <p>The target table is a replica table, which is used only for update-anywhere replication. The replica target table receives changes from the master source table, and changes to the replica target table are propagated back to the master source table. A replica table is automatically registered as a source table.</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>TIMING</b>	<p>Specifies the type of timing (scheduling) that the Apply program uses to process the subscription set.</p> <p><b>*INTERVAL</b> (default) The Apply program processes the subscription set at a specific time interval (for example, once a day).</p> <p><b>*EVENT</b> The Apply program processes the subscription set when a specific event occurs.</p> <p><b>*BOTH</b> The Apply program processes the subscription set either at a specific interval or when an event occurs, whichever occurs first.</p>
<b>EVENT</b>	<p>Specifies an event. The event that you enter must match an event name in the subscription events (IBMSNAP_SUBS_EVENT) table.</p> <p><b>*NONE</b> (default) No event is used.</p> <p><i>event-name</i> A unique character string that represents an event described in the IBMSNAP_SUBS_EVENT table.</p>
<b>INTERVAL</b>	<p>Specifies the time interval (weeks, days, hours, and minutes) from start time to start time between refreshes of the target copy. This is a two-part value. The first part is a number; the second part is the unit of time:</p> <p><b>*MIN</b> Minutes</p> <p><b>*HOUR</b> Hours</p> <p><b>*DAY</b> Days</p> <p><b>*WEEK</b> Weeks</p> <p>You can specify combinations of numbers with units of time. For example, ((2 *WEEK) (3 *DAY) (35 *MIN)) specifies a time interval of two weeks, three days, and 35 minutes. If you specify multiple occurrences of the same unit of time, the last occurrence is used.</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>ACTIVATE</b>	<p>Specifies whether the subscription set is active. The Apply program does not process this subscription set unless this parameter is set to *YES.</p> <p><b>*YES</b> (default) The subscription set is active.</p> <p><b>*NO</b> The subscription set is not active.</p>
<b>CRTTGTTBL</b>	<p>Specifies whether the target table (or view) is created.</p> <p><b>*YES</b> (default) Creates the target table (or view) if it does not exist. Otherwise, the existing table or view becomes the target, and the format of this existing table or view is checked if the value of the <b>CHKFMT</b> parameter is set to *YES. An additional index, with the values that you specified by the <b>UNIQUE</b> and <b>KEYCOL</b> parameters, is created for a target table if no such index currently exists. The command fails if an existing target table contains rows that violate the conditions of the additional index.</p> <p><b>*NO</b> Does not create the target table or view. You must create the table or view with the correct attributes before starting the Apply program.</p> <p>If the table or view exists and you set <b>CHKFMT</b> to *YES, the <b>ADDDPRSUB</b> command ensures that the format of the existing table matches the subscription-set definition that you set. If <b>CHKFMT</b> is *NO, you must ensure that the format of the existing table matches the subscription-set definition.</p> <p><b>Important:</b> If the table or view already exists, DB2 DataPropagator for iSeries does not automatically journal changes to the existing object. You must start journaling outside of DB2 DataPropagator for iSeries.</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>CHKFMT</b>	<p>Specifies whether DB2 DataPropagator for iSeries checks the subscription set and the target table to ensure that the columns match. This parameter is ignored if the <b>CRITGTTBL</b> parameter is *YES; this parameter is also ignored if the <b>CRITGTTBL</b> parameter is set to *NO and the target table does not exist.</p> <p><b>*YES</b> (default) DB2 DataPropagator for iSeries verifies that the columns defined for this subscription set match the columns in the target table. This command fails if a mismatch is detected.</p> <p><b>*NO</b> DB2 DataPropagator for iSeries ignores the differences between the subscription set and the existing target table. You must ensure that the target table is compatible with the subscription set.</p>
<b>CAPCTLLIB</b>	<p>Specifies the Capture schema, which is the name of the library in which the Capture control tables reside. These Capture control tables process the source for this subscription set.</p> <p><b>ASN</b> (default) The Capture control tables reside in the ASN library.</p> <p><i>library-name</i> The name of a library that contains the Capture control tables. This is the library in which the source table was registered.</p>
<b>TGTCCLIB</b>	<p>Specifies the target control library.</p> <p><b>*CAPCTLLIB</b> (default) The target control library is the same library in which the Capture control tables reside.</p> <p><i>library-name</i> The name of a library that contains the target control tables.</p> <p>If you are using a target table as a source for another subscription set (such as an external CCD table), this parameter value is the Capture schema when this table is used as a source.</p>
<b>FEDSVR</b>	<p>Specifies whether a federated database system is the source for this subscription set.</p> <p><b>*NONE</b> (default) The source server is not a federated database system.</p> <p><i>server-name</i> The name of the federated database system for this subscription set (for non-DB2 relational sources).</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>CMTCNT</b>	<p>Specifies the commitment count, which is the number of transactions that the Apply program processes before a commit.</p> <p><b>*DEFAULT</b> (default) The command determines the value to use. If the <b>TGTTYPE</b> is set to <b>*REPLICA</b>, then the <b>CMTCNT</b> is zero (0). If the <b>TGTTYPE</b> is anything other than <b>*REPLICA</b>, the <b>CMTCNT</b> is null.</p> <p><b>*NULL</b> The subscription set is read-only. The Apply program will fetch answer sets for the subscription-set members one member at a time, until all data has been processed and then will issue a single commit for the entire subscription set.</p> <p><i>num-transactions</i> Specifies the number of transactions processed before the Apply program commits the changes. This parameter is valid only if the <b>TGTTYPE</b> parameter is set to <b>*REPLICA</b>.</p> <p><b>Important:</b> A value of 1 is the same as the value of 0.</p>
<b>TGTKEYCHG</b>	<p>Specifies how the Apply program handles updates when changes occur in source columns that are part of the target key columns for the target table. This parameter works in conjunction with the <b>USEDELINS</b> parameter on the <b>ADDDPRREG</b> command:</p> <ul style="list-style-type: none"> <li>• If <b>USEDELINS</b> is YES and <b>TGTKEYCHG</b> is YES, updates are not allowed.</li> <li>• If <b>USEDELINS</b> is YES and <b>TGTKEYCHG</b> is NO, updates become delete and insert pairs.</li> <li>• If <b>USEDELINS</b> is NO and <b>TGTKEYCHG</b> is YES, the Apply program handles this condition with special logic.</li> <li>• If <b>USEDELINS</b> is NO and <b>TGTKEYCHG</b> is NO, the Apply program processes the changes as normal updates.</li> </ul> <p><b>*NO</b> (default) Updates to the source table are staged by the Capture program and processed by the Apply program to the target table.</p> <p><b>*YES</b> The Apply program updates the target table based on the before images of the target key column, meaning that the Apply program changes the predicate to the old values instead of the new.</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>COLUMN</b>	<p>Specifies the columns to be included in the target table. The column names must be unqualified. Choose the column names from the list of column names that you specified with the <b>CAPCOL</b> parameter when you registered the source table.</p> <p>If you set the <b>IMAGE</b> parameter to <b>*BOTH</b> when registering this table, you can specify before-image column names. The before-image column names are the original column names with a prefix. This prefix is the character that you specified in the <b>PREFIX</b> parameter of the <b>ADDDPRREG</b> command.</p> <p><b>*ALL</b> (default) All of the columns that you registered in the source are included in the target table.</p> <p><b>*NONE</b> No columns from the source table are included in the target table. You can use <b>*NONE</b> when you want only computed columns in the target table. This value is required if the <b>CALCCOL</b> parameter contains summary functions but no <b>GROUP BY</b> is performed.</p> <p><i>column-name</i> The names of up to 300 source columns that you want to include in the target table. Separate the column names with spaces.</p>
<b>UNIQUE</b>	<p>Specifies whether the target table has unique keys as indicated by the <b>KEYCOL</b> parameter.</p> <p><b>*YES</b> (default) The target table supports one net change per key; only one row exists in the target table for that key regardless of how many changes are made to the key.</p> <p>This value specifies that the table contains current data rather than a history of changes to the data. A condensed table includes no more than one row for each primary key value in the table and can be used to supply current information for a refresh.</p> <p><b>*NO</b> The target table supports multiple changes per key. The changes are appended to the target table.</p> <p>This value specifies that the table contains a history of changes to the data rather than current data. A non-condensed table includes more than one row for each key value in the table and can be used to supply a history of changes to the data. A non-condensed table cannot supply current data for a refresh.</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
KEYCOL	<p>Specifies columns that describe the key of the target table. The column names must be unqualified. For *POINTINTIME, *REPLICA, and *USERCOPY target tables (as specified on the TGTTYPE parameter), you must identify one or more columns as the target key for the target table. This target key is used by the Apply program to identify each unique row that changes during change-capture replication.</p> <p><b>*SRCTBL</b> (default)</p> <p>The key columns in the target table are the same as those in the source table. The ADDDPRREG command uses the key that is specified in the source table if the source table is keyed. The following key columns are used:</p> <ul style="list-style-type: none"><li>• Key columns that you defined through DDS when creating the table with the Create Physical File (CRTPF) command</li><li>• Primary and unique keys that you defined with the CREATE TABLE and ALTER TABLE SQL statements</li><li>• Unique keys that you defined with the CREATE INDEX SQL statements</li></ul> <p>If you use a column as a key more than once and with different ordering, the target table key is defined with ascending order.</p> <p><b>*RRN</b></p> <p>The key column in the target table is the IBMQSQ_RRN column. The target table is created with an IBMQSQ_RRN column, and this column is used as the key. When the Apply program runs, if the source table is a user table and the target table is a point-in-time or user copy, the IBMQSQ_RRN column in the target table is updated with the relative record number of the associated record in the source table. Otherwise, the IBMQSQ_RRN column in the target table is updated with the value of the IBMQSQ_RRN column in the source table.</p> <p><b>*NONE</b></p> <p>The target copy does not contain a target key. You cannot specify *NONE if the target table type is *POINTINTIME, *REPLICA, or *USERCOPY.</p> <p><i>column-name</i></p> <p>The names of the target columns that you want to use as the target key columns. You can specify up to 120 column names. Separate the column names with spaces.</p>



Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
TGTCOL	<p>Specifies the new names for all the columns that the Apply program updates in the target table. These names override the column names taken from the source table. The column names must be unqualified. If you specified a value of *NONE for the <b>COLUMN</b> parameter, do not use this parameter.</p> <p>Use this parameter to give more meaningful names to the target table columns. Specify each source column name and the name of the corresponding column on the target table.</p> <p><b>*COLUMN</b> (default) The target columns are the same as the columns you specified in the <b>COLUMN</b> parameter.</p> <p><i>column-name</i> The column names from the source table that you want to change at the target. You can list up to 300 column names.</p> <p><i>new-name</i> The new names of the target columns. You can list up to 300 new column names. If you do not use this parameter, the name of the column on the target table will be the same as the source column name.</p>
CALCCOL	<p>Specifies the list of user-defined or calculated columns in the target table. The column names must be unqualified. Enclose each column name and expression pair in parenthesis.</p> <p>You must specify a column name for each SQL expression. If you want to define any column as an SQL expression without a GROUP BY statement, you must set the <b>COLUMN</b> parameter to *NONE.</p> <p><b>*NONE</b> (default) No user-defined or calculated columns are included in the target table.</p> <p><i>column-name</i> The column names of the user-defined or calculated columns in the target table. You can list up to 100 column names.</p> <p><i>expression</i> The expressions for the user-defined or calculated columns in the target table. You can list up to 100 SQL column expressions.</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
ADDREG	<p>Specifies whether the target table is automatically registered as a source table. Use this parameter to register CCD target type tables.</p> <p><b>*NO</b> (default)</p> <p>The target table is not registered as a source table. DB2 DataPropagator for iSeries ignores this parameter value if the target type is *REPLICA. Replica target tables are always automatically registered as source tables.</p> <p><b>*YES</b></p> <p>The target table is registered as a source table. This command fails if you already registered the target table.</p> <p>Do not set this parameter to *YES if the target table type is *USERCOPY, *POINTINTIME, *BASEAGR, or *CHANGEAGR.</p> <p>If you set the <b>CRTTGTTBL</b> parameter to *NO, you must create the target table before attempting to register it as a source.</p>
ROWSLT	<p>Specifies the predicates to be placed in an SQL WHERE clause. The Apply program uses these predicates to determine which rows in the change-data (CD) table of the source to apply to the target table. Use this parameter if you want only a subset of the source changes to be replicated to the target table.</p> <p><b>*ALL</b> (default)</p> <p>The Apply program applies all changes in the CD table to the target table.</p> <p><i>WHERE-clause</i></p> <p>The SQL WHERE clause that specifies which rows from the CD table the Apply program applies to the target table. Do not include the WHERE keyword; it is implied on this parameter. This WHERE clause must be valid on the data server you are using to run the clause.</p> <p><b>Note:</b> The WHERE clause on this parameter is unrelated to any WHERE clauses specified on the SQLBEFORE or SQLAFTER parameters.</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>MAXSYNCH</b>	<p>Specifies the maximum synch minutes. This parameter is the time-threshold limit used to regulate the amount of change data that the Capture and Apply programs process during a subscription cycle. You can specify the time-threshold limit using a two-part value. The first part is a number; the second part is the unit of time:</p> <p><b>*MIN</b> Minutes</p> <p><b>*HOUR</b> Hours</p> <p><b>*DAY</b> Days</p> <p><b>*WEEK</b> Weeks</p> <p>You can specify combinations of numbers with units of time. For example, ((1 *WEEK) (2 *DAY) (35 *MIN)) specifies a time interval of one week, two days, and 35 minutes. If you specify multiple occurrences of the same unit of time, the last occurrence is used.</p> <p>The default is zero (0), which indicates that all of the change data is to be applied.</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
SQLBEFORE	<p>Specifies the SQL statements that run before the Apply program refreshes the target table. This parameter has three elements:</p> <p>Element 1: SQL code</p> <p><b>*NONE</b> (default) No SQL statement is specified.</p> <p><i>SQL-statement</i> The SQL statement that you want to run. Ensure that the syntax of the SQL statement is correct. DB2 DataPropagator for iSeries does not validate the syntax. In addition, you must use the proper SQL naming conventions. SQL file references must be in the form of LIBRARY.FILE instead of the system naming convention (LIBRARY/FILE). You can specify up to three SQL statements.</p> <p>Element 2: Server to run on</p> <p><b>*TGTSVR</b> (default) The SQL statement runs at the target server on which the target table is located.</p> <p><b>*SRCSVR</b> The SQL statement runs at the Capture control server on which the source table is located.</p> <p>Element 3: Allowed SQLSTATE values</p> <p><b>*NONE</b> (default) Only an SQLSTATE value of 00000 is considered successful.</p> <p><i>SQL-states</i> A list of one to ten allowable SQLSTATE values. Separate the SQLSTATE values with spaces. An SQLSTATE value is a five-digit hexadecimal number ranging from 00000 to FFFFF.</p> <p>The SQL statement is successful if it completes with an SQLSTATE value of 00000 or with one of the allowable SQLSTATE values that you listed.</p>

Table 31. ADDDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>SQLAFTER</b>	<p>Specifies SQL statements that run after the Apply program refreshes the target table. This parameter has three elements:</p> <p>Element 1: SQL code</p> <p><b>*NONE</b> (default) No SQL statement is specified.</p> <p><i>SQL-statement</i> The SQL statement that you want to run. Ensure that the syntax of the SQL statement is correct. DB2 DataPropagator for iSeries does not validate the syntax. In addition, you must use the proper SQL naming conventions. SQL file references must be in the form of LIBRARY.FILE instead of the system naming convention (LIBRARY/FILE). You can specify up to three SQL statements.</p> <p>Element 2: Server to run on</p> <p><b>*TGTSVR</b> (default) The SQL statement runs at the target server on which the target table is located.</p> <p>Element 3: Allowed SQLSTATE values</p> <p><b>*NONE</b> (default) Only an SQLSTATE value of 00000 is considered successful.</p> <p><i>SQL-states</i> A list of one to ten allowable SQLSTATE values. Separate the SQLSTATE values with spaces. An SQLSTATE value is a five-digit hexadecimal number ranging from 00000 to FFFFF.</p> <p>The SQL statement is successful if it completes with an SQLSTATE value of 00000 or with one of the allowable SQLSTATE values that you listed.</p>

## Examples for ADDDPRSUB

The following examples illustrate how to use the **ADDDPRSUB** command.

### Example 1

To create a subscription set named **SETHR** under the **AQHR** Apply qualifier:

```
ADDDPRSUB APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/EMPLOYEE)
          TGTTBL(TGTLIB/TGTEMPL)
```

This subscription set, which contains one subscription-set member, replicates data from the registered source table named **EMPLOYEE** under the **HR** library to the target table named **TGTEMPL** under the **TGTLIB** library.

## ADDDPRSUB

### Example 2

To create a subscription set named SETHR with only two columns, EMPNO (the key) and NAME, from the registered source table named EMPLOYEE and replicate these columns to an existing target table named TGTEMPL:

```
ADDDPRSUB APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/EMPLOYEE)
          TGTTLB(TGTLIB/TGTEMPL) CRTTGTTLB(*NO) COLUMN(EMPNO NAME) KEYCOL(EMPNO)
```

### Example 3

To create a subscription set named SETHR with data from the registered source table named EMPLOYEE and to replicate this data to a replica type target table named TGTREPL:

```
ADDDPRSUB APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/EMPLOYEE)
          TGTTLB(TGTLIB/TGTREPL) TGTTYPE(*REPLICA)
```

### Example 4

To create a subscription set named NOMEM with no subscription-set members:

```
ADDDPRSUB APYQUAL(AQHR) SETNAME(NOMEM) SRCTBL(*NONE) TGTTLB(*NONE)
```

### Related tasks:

- Chapter 4, “Subscribing to sources” on page 63

---

## ADDDPRSUBM: Adding a DPR subscription-set member (OS/400)

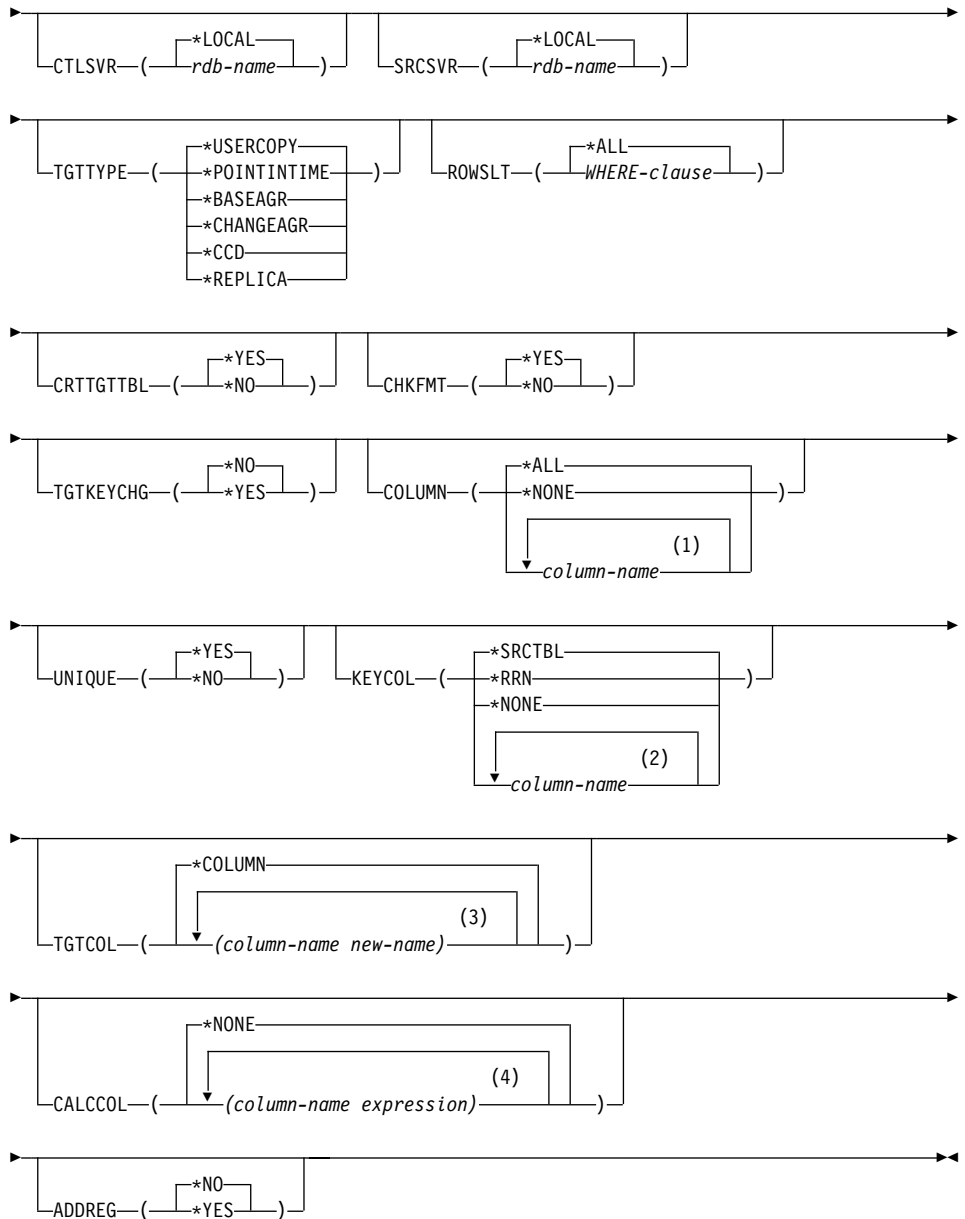
Use the Add DPR subscription-set member (**ADDDPRSUBM**) command to add a member to an existing subscription set. You can create the subscription set with the **ADDDPRSUB** command, with the system commands on UNIX, Windows, or z/OS, or through the Replication Center. All the source tables in the subscription set must already be journaled and must already be registered before you can use this command.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

***To add a member to a subscription set using the ADDDPRSUBM command:***

```
►►—ADDDPRSUBM—APYQUAL—(—apply-qualifier—)—SETNAME—(—set-name—)—►►
►►—SRCTBL—(—library-name/file-name—)—TGTTLB—(—library-name/file-name—)—►►
```



### Notes:

- 1 You can specify up to 300 column names.
- 2 You can specify up to 120 column names.
- 3 You can specify up to 300 column names.

4 You can specify up to 100 column names and expressions.

Table 32. ADDDPRSUBM command parameter definitions for OS/400

Parameter	Definition and prompts
<b>APYQUAL</b>	Specifies the Apply qualifier that identifies which Apply program processes this subscription set. Subscription sets under an Apply qualifier run in a separate job. This parameter is required.  <i>apply-qualifier</i> The name of the Apply qualifier.
<b>SETNAME</b>	Specifies the name of the subscription set. This parameter is required.  <i>set-name</i> The name of the subscription set. The subscription-set name that you enter must be unique for the specified Apply qualifier or the <b>ADDDPRSUBM</b> command produces an error. Because the Apply program handles the set of target tables as a group, when one target table fails for any reason, the entire set fails.
<b>SRCTBL</b>	Specifies the name of the table that is the source for this subscription-set member. You must register this table at the Capture control server before this table can become a member of a subscription set. This parameter is required.  <i>library-name/file-name</i> The qualified name of the source table.
<b>TGTTBL</b>	Specifies the name of the target table for this subscription-set member. The target table is automatically created if you set the <b>CRTTGTTBL</b> parameter to *YES and the target table does not already exist. This parameter is required.  <i>library-name/file-name</i> The qualified name of the target table.
<b>CTLSVR</b>	Specifies the relational database name of the system that contains the Apply control tables.  <b>*LOCAL</b> (default) The Apply control tables reside locally (on the machine from which you are running the <b>ADDDPRSUBM</b> command).  <i>rdb-name</i> The name of the relational database where the Apply control tables reside. You can use the Work with RDB Directory Entries ( <b>WRKRDBDIRE</b> ) command to find this name.



Table 32. ADDDPRSUBM command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>SRCSVR</b>	<p>Specifies the relational database name of the system that contains the Capture control tables.</p> <p><b>*LOCAL</b> (default) The source table is registered on the local machine (the machine from which you are running the <b>ADDDPRSUBM</b> command).</p> <p><i>rdb-name</i> The name of the relational database where the Capture control tables reside. You can use the Work with RDB Directory Entries (<b>WRKRDBDI</b>) command to find this name.</p>
<b>TGTTYPE</b>	<p>Specifies the target table type. These are DB2 replication terms that describe the contents of the target table. After you create a target table as one of these types, you can use this parameter value on the <b>SRCTBL</b> parameter of the Add DPR Registration (<b>ADDDPRREG</b>) command to register this target table as a source table.</p> <p><b>*USERCOPY</b> (default) The target table is a user copy, which is a target table with content that matches all or part of the content of a source table. A user copy is handled like a point-in-time copy but does not contain any of the DB2 DataPropagator for iSeries system columns that are present in the point-in-time target table.</p> <p>This value is not valid when a value of <b>*RRN</b> is specified on the <b>KEYCOL</b> parameter.</p> <p>The table that you specified with the <b>SRCTBL</b> parameter must be one of the following types: user database, point-in-time copy, or consistent-change data (CCD).</p> <p><b>Important:</b> If the target table already exists, DB2 DataPropagator for iSeries does not automatically journal changes to it. You must start journaling outside of DB2 DataPropagator for iSeries.</p>

Table 32. ADDDPRSUBM command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
TGTTYPE (continued)	<p><b>*POINTINTIME</b></p> <p>The target table is a point-in-time copy. A point-in-time copy is a target table with content that matches all or part of the content of the source table and includes the DB2 DataPropagator for iSeries system column (IBMSNAP_LOGMARKER), which identifies when a particular row was inserted or updated at the Capture control server.</p> <p><b>*BASEAGR</b></p> <p>The target table is a base aggregate copy, which is a target table that contains data that is aggregated (calculated) from a source table. The source table for a base aggregate target must be either a user table or a point-in-time table. This target table must contain the IBMSNAP_HLOGMARKER and IBMSNAP_LLOGMARKER system timestamp columns.</p> <p><b>*CHANGEAGR</b></p> <p>The table is a change aggregate copy, which is a target table that contains data that is aggregated (calculated) based on the contents of a change-data (CD) table. This target table is created with the IBMSNAP_HLOGMARKER and IBMSNAP_LLOGMARKER system timestamp columns.</p> <p><b>*CCD</b></p> <p>The table is a consistent-change data (CCD) table, which is a target table created from a join of data in the change-data (CD) table and the unit-of-work (UOW) table. A CCD table provides transaction-consistent data for the Apply program and must include the following columns:</p> <ul style="list-style-type: none"><li>• IBMSNAP_INTENTSEQ</li><li>• IBMSNAP_OPERATION</li><li>• IBMSNAP_COMMITSEQ</li><li>• IBMSNAP_LOGMARKER</li></ul> <p><b>*REPLICA</b></p> <p>The target table is a replica table, which is used only for update-anywhere replication. The replica target table receives changes from the master source table, and changes to the replica target table are propagated back to the master source table. A replica table is automatically registered as a source table.</p>

Table 32. ADDDPRSUBM command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>ROWSLT</b>	<p>Specifies the predicates to be placed in an SQL WHERE clause. The Apply program uses these predicates to determine which rows in the change-data (CD) table of the source to apply to the target table. Use this parameter if you want only a subset of the source changes to be replicated to the target table.</p> <p><b>*ALL</b> (default) The Apply program applies all changes in the CD table to the target table.</p> <p><i>WHERE-clause</i> The SQL WHERE clause that specifies which rows from the CD table the Apply program applies to the target table. Do not include the WHERE keyword; it is implied on this parameter. This WHERE clause must be valid on the data server you are using to run the clause.</p> <p><b>Note:</b> The WHERE clause on this parameter is unrelated to any WHERE clauses specified on the SQLBEFORE or SQLAFTER parameters.</p>
<b>CRTTGTTBL</b>	<p>Specifies whether the target table (or view) is created.</p> <p><b>*YES</b> (default) Creates the target table (or view) if it does not exist. Otherwise, the existing table or view becomes the target, and the format of this existing table or view is checked if the value of the <b>CHKFMT</b> parameter is set to *YES. An additional index, with the values that you specified by the <b>UNIQUE</b> and <b>KEYCOL</b> parameters, is created for a target table if no such index currently exists. The command fails if an existing target table contains rows that violate the conditions of the additional index.</p> <p><b>*NO</b> Does not create the target table or view. You must create the table or view with the correct attributes before starting the Apply program.</p> <p>If the table or view exists and you set <b>CHKFMT</b> to *YES, the <b>ADDDPRSUBM</b> command ensures that the format of the existing table matches the subscription-set definition that you set. If <b>CHKFMT</b> is *NO, you must ensure that the format of the existing table matches the subscription-set definition.</p> <p><b>Important:</b> If the table or view already exists, DB2 DataPropagator for iSeries does not automatically journal changes to the existing object. You must start journaling outside of DB2 DataPropagator for iSeries.</p>

Table 32. ADDDPRSUBM command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>CHKFMT</b>	<p>Specifies whether DB2 DataPropagator for iSeries checks the definition of the subscription-set member against the existing target table to ensure that the columns match. This parameter is ignored if the <b>CRTTGTTBL</b> parameter is *YES; this parameter is also ignored if the <b>CRTTGTTBL</b> parameter is set to *NO and the target table does not exist.</p> <p><b>*YES</b> (default) DB2 DataPropagator for iSeries verifies that the columns defined for this subscription-set member match the columns in the target table. This command fails if a mismatch is detected.</p> <p><b>*NO</b> DB2 DataPropagator for iSeries ignores differences between the subscription-set member and the existing target table. You must ensure that the target table is compatible with the subscription-set member.</p>
<b>TGTKEYCHG</b>	<p>Specifies how the Apply program handles updates when changes occur in source columns that are part of the target key columns for the target table. This parameter works in conjunction with the <b>USEDELINS</b> parameter on the <b>ADDDPRREG</b> command:</p> <ul style="list-style-type: none"> <li>• If <b>USEDELINS</b> is YES and <b>TGTKEYCHG</b> is YES, updates are not allowed.</li> <li>• If <b>USEDELINS</b> is YES and <b>TGTKEYCHG</b> is NO, updates become delete and insert pairs.</li> <li>• If <b>USEDELINS</b> is NO and <b>TGTKEYCHG</b> is YES, the Apply program handles this condition with special logic.</li> <li>• If <b>USEDELINS</b> is NO and <b>TGTKEYCHG</b> is NO, the Apply program processes the changes as normal updates.</li> </ul> <p><b>*NO</b> (default) Updates to the source table are staged by the Capture program and processed by the Apply program to the target table.</p> <p><b>*YES</b> The Apply program updates the target table based on the before images of the target key column, meaning that the Apply program changes the predicate to the old values instead of the new.</p>

Table 32. ADDDPRSUBM command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>COLUMN</b>	<p>Specifies the columns to be included in the target table. The column names must be unqualified. Choose the column names from the list of column names that you specified on the <b>CAPCOL</b> parameter when you registered the source table.</p> <p>If you set the <b>IMAGE</b> parameter to <b>*BOTH</b> when registering this table, you can specify before-image column names. The before-image column names are the original column names with a prefix. This prefix is the character that you specified in the <b>PREFIX</b> parameter of the <b>ADDDPRREG</b> command.</p> <p><b>*ALL</b> (default) All of the columns that you registered in the source are included in the target table.</p> <p><b>*NONE</b> No columns from the source table are included in the target table. You can use <b>*NONE</b> when you want only computed columns in the target table. This value is required if the <b>CALCCOL</b> parameter contains summary functions but no <b>GROUP BY</b> is performed.</p> <p><i>column-name</i> The names of up to 300 source columns that you want to include in the target table. Separate the column names with spaces.</p>
<b>UNIQUE</b>	<p>Specifies whether the target table has unique keys as indicated by the <b>KEYCOL</b> parameter.</p> <p><b>*YES</b> (default) The target table supports one net change per key; only one row exists in the target table for that key regardless of how many changes are made to the key.</p> <p>This value specifies that the table contains current data rather than a history of changes to the data. A condensed table includes no more than one row for each primary key value in the table and can be used to supply current information for a refresh.</p> <p><b>*NO</b> The target table supports multiple changes per key. The changes are appended to the target table.</p> <p>This value specifies that the table contains a history of changes to the data rather than current data. A non-condensed table includes more than one row for each key value in the table and can be used to supply a history of changes to the data. A non-condensed table cannot supply current data for a refresh.</p>

Table 32. ADDDPRSUBM command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
KEYCOL	<p>Specifies columns that describe the key of the target table. The column names must be unqualified. For *POINTINTIME, *REPLICA, and *USERCOPY target tables (as specified on the TGTTYPE parameter), you must identify one or more columns as the target key for the target table. This target key is used by the Apply program to identify each unique row that changes during change-capture replication.</p> <p><b>*SRCTBL</b> (default)</p> <p>The key columns in the target table are the same as those in the source table. The ADDDPRREG command uses the key that is specified in the source table if the source table is keyed. The following key columns are used:</p> <ul style="list-style-type: none"><li>• Key columns that you defined through DDS when creating the table with the Create Physical File (CRTPF) command</li><li>• Primary and unique keys that you defined with the CREATE TABLE and ALTER TABLE SQL statements</li><li>• Unique keys that you defined with the CREATE INDEX SQL statements</li></ul> <p>If you use a column as a key more than once and with different ordering, the target table key is defined with ascending order.</p> <p><b>*RRN</b></p> <p>The key column in the target table is the IBMQSQ_RRN column. The target table is created with an IBMQSQ_RRN column, and this column is used as the key. When the Apply program runs, if the source table is a user table and the target table is a point-in-time or user copy, the IBMQSQ_RRN column in the target table is updated with the relative record number of the associated record in the source table. Otherwise, the IBMQSQ_RRN column in the target table is updated with the value of the IBMQSQ_RRN column in the source table.</p> <p><b>*NONE</b></p> <p>The target copy does not contain a target key. You cannot specify *NONE if the target table type is *POINTINTIME, *REPLICA, or *USERCOPY.</p> <p><i>column-name</i></p> <p>The names of the target columns that you want to use as the target key columns. You can specify up to 120 column names. Separate the column names with spaces.</p>

Table 32. ADDDPRSUBM command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
TGTCOL	<p>Specifies the new names for all the columns that the Apply program updates in the target table. These names override the column names taken from the source table. The column names must be unqualified. If you specified a value of *NONE for the <b>COLUMN</b> parameter, do not use this parameter.</p> <p>Use this parameter to give more meaningful names to the target table columns. Specify each source column name and the name of the corresponding column on the target table.</p> <p><b>*COLUMN</b> (default) The target columns are the same as the columns you specified in the <b>COLUMN</b> parameter.</p> <p><i>column-name</i> The column names from the source table that you want to change at the target. You can list up to 300 column names.</p> <p><i>new-name</i> The new names of the target columns. You can list up to 300 new column names. If you do not use this parameter, the name of the column on the target table will be the same as the source column name.</p>
CALCCOL	<p>Specifies the list of user-defined or calculated columns in the target table. The column names must be unqualified. Enclose each column name and expression pair in parenthesis.</p> <p>You must specify a column name for each SQL expression. If you want to define any column as an SQL expression without a GROUP BY statement, you must set the <b>COLUMN</b> parameter to *NONE.</p> <p><b>*NONE</b> (default) No user-defined or calculated columns are included in the target table.</p> <p><i>column-name</i> The column names of the user-defined or calculated columns in the target table. You can list up to 100 column names.</p> <p><i>expression</i> The expressions for the user-defined or calculated columns in the target table. You can list up to 100 SQL column expressions.</p>

Table 32. ADDDPRSUBM command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
ADDREG	<p>Specifies whether the target table is automatically registered as a source table. Use this parameter to register CCD target type tables.</p> <p><b>*NO (default)</b></p> <p>The target table is not registered as a source table. DB2 DataPropagator for iSeries ignores this parameter value if the target type is *REPLICA. Replica target tables are always automatically registered as source tables.</p> <p><b>*YES</b></p> <p>The target table is registered as a source table. This command fails if you already registered the target table.</p> <p>Do not set this parameter to *YES if the target table type is *USERCOPY, *POINTINTIME, *BASEAGR, or *CHANGEAGR.</p> <p>If you set the <b>CRTTGTTBL</b> parameter to *NO, you must create the target table before attempting to register it as a source.</p>

Examples for ADDDPRSUBM

The following examples illustrate how to use the **ADDDPRSUBM** command.

Example 1

To add a subscription-set member to a subscription set named SETHR under the AQHR Apply qualifier:

```
ADDDPRSUBM APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/YTD TAX) TGTTBL(TGTHR/TGTTAX)
```

Example 2

To add a subscription-set member with only two columns, AMOUNT and NAME, from the registered source table named YTD TAX and to replicate these columns to an existing target table named TGTTAX:

```
ADDDPRSUBM APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/YTD TAX) TGTTBL(TGTLIB/TGTTAX)
CRTTGTTBL(*NO) COLUMN(AMOUNT NAME) CHKfmt(*YES)
```

This command verifies that the AMOUNT and NAME columns defined for this subscription-set member match the columns in the target table.

Example 3

To add a subscription-set member to subscription set named SETHR and to replicate this data to a consistent-change data target table named TGTYTD:

```
ADDDPRSUBM APYQUAL(AQHR) SETNAME(SETHR) SRCTBL(HR/YTD TAX) TGTTBL(TGTLIB/TGTYTD)
TGTTTYPE(*CCD) ADDREG (*YES)
```

This command registers the target table as a source table for DB2 DataPropagator for iSeries.



**Related tasks:**

- Chapter 4, “Subscribing to sources” on page 63

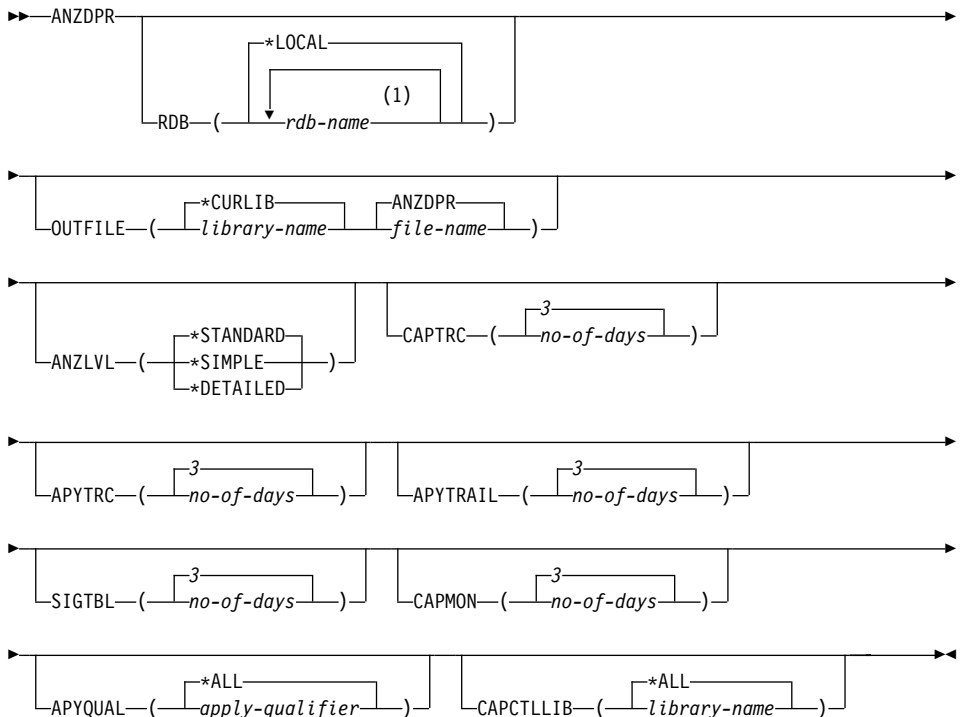
## ANZDPR: Operating the Analyzer (OS/400)

Use the Analyze DPR (ANZDPR) command to analyze a failure from a Capture or Apply program, to verify the setup of your replication configuration, or to obtain problem diagnosis and performance tuning information. Run this command after you set up your replication configuration.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

***To operate the Analyzer using the ANZDPR command:***



Notes:

1     You can specify up to 10 databases.

Table 33. ANZDPR command parameter definitions for OS/400

Parameter	Definition and prompts
RDB	<p>Specifies the databases to be analyzed.</p> <p><b>*LOCAL</b> (default) The database on your local system.</p> <p><i>rdb-name</i> The RDB Directory Entry name, which indicates the database.</p> <p>You can enter up to 10 databases. If you want to analyze multiple databases including the database on your local system, make sure that *LOCAL is the first entry in the list. Also, verify that you can connect to all these databases from your current system.</p>
OUTFILE	<p>Specifies the library and file name used to store the analyzer output. This command writes the output to an HTML file.</p> <p><b>*CURLIB</b> (default) The current library.</p> <p><i>library-name</i> The name of the library.</p> <p><b>ANZDPR</b> (default) The output is written to an HTML file named ANZDPR.</p> <p><i>file-name</i> The name of the HTML output file.</p> <p>If the file name already exists, the file is overwritten. If the file name does not exist, the command creates the file with attributes of RCDLEN(512) and SIZE(*NOMAX).</p>

Table 33. ANZDPR command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>ANZLVL</b>	<p>Specifies the level of analysis to be reported. The level of analysis can be:</p> <p><b>*STANDARD</b> (default) Generates a report that includes the contents of the control tables as well as Capture and Apply program status information.</p> <p><b>*SIMPLE</b> Generates the information in the standard report but excludes subcolumn details. Use this option if you want to generate a smaller report that requires less system resources.</p> <p><b>*DETAILED</b> Generates a report with the most complete analysis. The detailed report includes the information from the standard report in addition to subscription set information.</p>
<b>CAPTRC</b>	<p>Specifies the date range (0 to 30 days) of entries to be reported from the Capture trace (IBMSNAP_CAPTRACE) table. The default is 3.</p> <p><i>no-of-days</i> The number of days to be reported.</p>
<b>APYTRC</b>	<p>Specifies the date range (0 to 30 days) of entries to be reported from the Apply trace (IBMSNAP_APPLYTRACE) table. The default is 3.</p> <p><i>no-of-days</i> The number of days to be reported.</p>
<b>APYTRAIL</b>	<p>Specifies the date range (0 to 30 days) of entries to be reported from the Apply trail (IBMSNAP_APPLYTRAIL) table. The default is 3.</p> <p><i>no-of-days</i> The number of days to be reported.</p>
<b>SIGTBL</b>	<p>Specifies the date range (0 to 30 days) of entries to be reported from the signal (IBMSNAP_SIGNAL) table. The default is 3.</p> <p><i>no-of-days</i> The number of days to be reported.</p>
<b>CAPMON</b>	<p>Specifies the date range (0 to 30 days) of entries to be reported from the Capture monitor (IBMSNAP_CAPMON) table. The default is 3.</p> <p><i>no-of-days</i> The number of days to be reported.</p>

Table 33. ANZDPR command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
APYQUAL	<p>Specifies the Apply qualifiers to be analyzed.</p> <p><b>*ALL</b> (default) All Apply qualifiers are analyzed.</p> <p><i>apply-qualifier</i> The name of the Apply qualifier to be analyzed. You can enter up to 10 Apply qualifiers.</p>
CAPCTLLIB	<p>Specifies the Capture schemas, which are the names of the Capture control libraries that you want to analyze. You can analyze a specific Capture control library, or you can choose the default of <b>*ALL</b> to analyze all the Capture control libraries.</p> <p><b>*ALL</b> (default) All of the Capture control libraries will be analyzed.</p> <p><i>library-name</i> The name of the specific Capture control library that you want to analyze.</p>

Examples for ANZDPR

The following examples illustrate how to use the **ANZDPR** command.

Example 1

To run the Analyzer on both your local database and a remote database named RMTRDB1 using a standard level of analysis:

```
ANZDPR RDB(*LOCAL RMTRDB1) OUTFILE(MYLIB/ANZDPR) ANZLVL(*STANDARD) CAPTRC(1)
      APYTRC(1) APYTRAIL(1) SIGTBL(1) CAPMON(1) APYQUAL(*ALL)
```

This example generates one day of entries from the IBMSNAP\_CAPTRACE, IBMSNAP\_APPLYTRACE, IBMSNAP\_APPLYTRAIL, IBMSNAP\_SIGNAL, and IBMSNAP\_CAPMON tables for all Apply qualifiers and writes the output to an HTML file named ANZDPR in the library called MYLIB.

Example 2

To run the Analyzer with all default values:

```
ANZDPR
```

Related reference:

- “asnanalyze: Operating the Analyzer (UNIX and Windows)” on page 305

## CHGDPRCAPA: Changing DPR Capture attributes (OS/400)

Use the Change DPR Capture Attributes (**CHGDPRCAPA**) command to change the global operating parameters that are used by the Capture program and are stored in the Capture parameters (IBMSNAP\_CAPPARMS) table. These parameter changes do not take effect until you perform one of the following actions:

- Issue an **INZDPRCAP** command.
- End and then restart the Capture program.

To change the behavior of a running Capture program, see “OVRDPRCAPA: Overriding DPR capture attributes (OS/400)” on page 414.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

### ***To change the DPR Capture attributes using the CHGDPRCAPA command:***

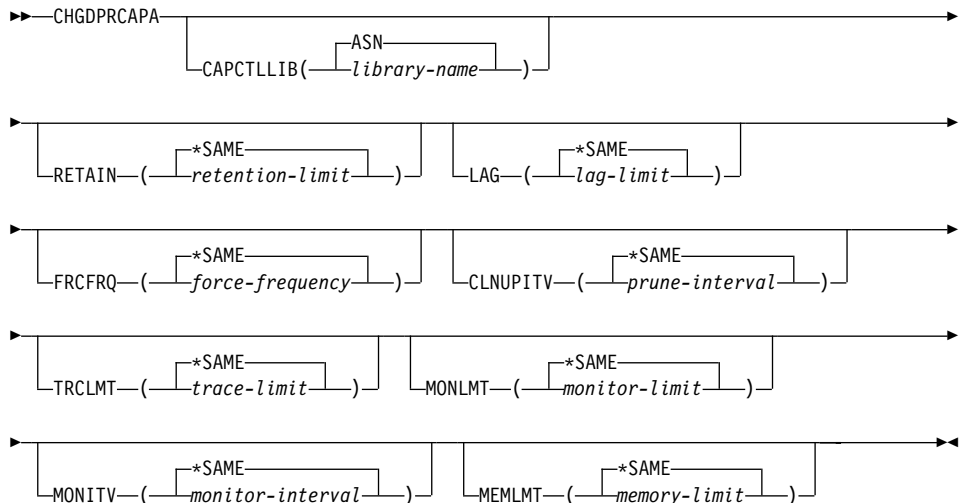


Table 34. CHGDPRCAPA command parameter definitions for OS/400

Parameter	Definition and prompts
CAPCTLLIB	Specifies the Capture schema, which is the name of the library in which the Capture control tables reside.
	ASN (default) The Capture control tables are in the ASN library.
	library-name The name of a library that contains the Capture control tables.
RETAIN	Specifies the new retention limit, which is the number of minutes that data is retained in the change-data (CD), unit-of-work (UOW), signal (IBMSNAP_SIGNAL), and Apply qualifier cross-reference (IBMSNAP_AUTHTKN) tables before this data is removed. This value is stored in the RETENTION_LIMIT column of the Capture parameters (IBMSNAP_CAPPARMS) table.
	This value works with the CLNUPITV parameter value. When the CLNUPITV value is reached, the CD, UOW, IBMSNAP_SIGNAL, and IBMSNAP_AUTHTKN data is removed if this data is older than the retention limit.
	Ensure that the Apply intervals are set to copy changed information before the data reaches this RETAIN parameter value to prevent inconsistent data in your tables. If the data becomes inconsistent, the Apply program performs a full refresh.
	The default is 10 080 minutes (seven days). The maximum is 35000000 minutes.
	*SAME (default) This value is not changed.
	retention-limit The new retention limit value.

Table 34. CHGDPRCAPA command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>LAG</b>	<p>Specifies the new lag limit, which is the number of minutes that the Capture program can fall behind in processing before restarting. This value is stored in the LAG_LIMIT column of the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p>When the lag limit is reached (that is, when the timestamp of the journal entry is older than the current timestamp minus the lag limit), the Capture program initiates a cold start for the tables that it is processing for that journal. The Apply program then performs a full refresh to provide the Capture program with a new starting point.</p> <p>The default is 10 080 minutes (seven days). The maximum is 35000000 minutes.</p> <p><b>*SAME</b> (default) This value is not changed.</p> <p><i>lag-limit</i> The new lag limit value.</p>
<b>FRCFRQ</b>	<p>Specifies how often (from 30 to 600 seconds) the Capture program writes changes to the change-data (CD) and unit-of-work (UOW) tables. This value is stored in the COMMIT_INTERVAL column of the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p>The Capture program makes these changes available to the Apply program either when the buffers are filled or when the <b>FRCFRQ</b> time limit expires, whichever is sooner.</p> <p>Use this parameter to make changes more readily available to the Apply program on servers with a low rate of source table changes. The <b>FRCFRQ</b> parameter value is a global value used for all defined source tables. Setting the <b>FRCFRQ</b> value to a low number can affect system performance.</p> <p>The default is 30 seconds.</p> <p><b>*SAME</b> (default) This value is not changed.</p> <p><i>force-frequency</i> The new commit interval value, which is the number of seconds that the Capture program keeps CD and UOW table changes in buffer space before making these changes available to the Apply program.</p>

Table 34. CHGDPRCAPA command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
CLNUPITV	<p>Specifies the maximum amount of time (in hours) before the Capture program prunes old records from the change-data (CD), unit-of-work (UOW), signal (IBMSNAP_SIGNAL), Capture monitor (IBMSNAP_CAPMON), Capture trace (IBMSNAP_CAPTRACE), and Apply qualifier cross-reference (IBMSNAP_AUTHTKN) tables.</p> <p>This parameter works in conjunction with the <b>RETAIN</b> parameter to control pruning of the CD, UOW, IBMSNAP_SIGNAL, and IBMSNAP_AUTHTKN tables, with the <b>MONLMT</b> parameter to control pruning of the IBMSNAP_CAPMON table, and with the <b>TRCLMT</b> parameter to control pruning of the IBMSNAP_CAPTRACE table. (Use the <b>STRDPRCAP</b> command to set the <b>RETAIN</b>, <b>MONLMT</b>, and <b>TRCLMT</b> parameters for a Capture program.)</p> <p>The value of this parameter is automatically converted from hours to seconds and is stored in the PRUNE_INTERVAL column of the Capture parameters (IBMSNAP_CAPPARMS) table. If the PRUNE_INTERVAL column is changed manually (not using the <b>CHGDPRCAPA</b> command), you might see changes due to rounding when you prompt using the F4 key.</p> <p><b>*SAME</b> (default) This Capture attribute value is not changed.</p> <p><i>prune-interval</i> The pruning interval expressed as a specific number of hours (1 to 100).</p>
TRCLMT	<p>Specifies the trace limit (in minutes). This value is stored in the TRACE_LIMIT column of the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p>The Capture programs prune any IBMSNAP_CAPTRACE rows that are older than the trace limit. The default is 10 080 minutes (seven days of trace entries).</p> <p><b>*SAME</b> (default) This value is not changed.</p> <p><i>trace-limit</i> The number of minutes of trace data kept in the IBMSNAP_CAPTRACE table after pruning.</p>



Table 34. CHGDPRCAPA command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>MONLMT</b>	<p>Specifies the monitor limit (in minutes). This value is stored in the MONITOR_LIMIT column of the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p>The Capture program prunes any IBMSNAP_CAPMON rows that are older than the monitor limit.</p> <p>The default is 10 080 minutes (seven days of monitor entries).</p> <p><b>*SAME</b> (default) This value is not changed.</p> <p><i>monitor-limit</i> The number of minutes of monitor data kept in the IBMSNAP_CAPMON table after pruning.</p>
<b>MONITV</b>	<p>Specifies how frequently (in seconds) the Capture program inserts rows into the Capture monitor (IBMSNAP_CAPMON) table. This value is stored in the MONITOR_INTERVAL column of the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p>The default is 300 seconds (five minutes).</p> <p><b>*SAME</b> (default) This value is not changed.</p> <p><i>monitor-interval</i> The number of seconds between row insertion into the IBMSNAP_CAPMON table. The monitor interval must be at least 120 seconds (two minutes). If you specify a number that is less than 120, this command automatically sets this parameter value to 120.</p>
<b>MEMLMT</b>	<p>Specifies the maximum size (in megabytes) of memory that the Capture journal job can use. This value is stored in the MEMORY_LIMIT column of the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p>The default is 32 megabytes.</p> <p><b>*SAME</b> (default) This value is not changed.</p> <p><i>memory-limit</i> The maximum number of megabytes for memory.</p>

## Examples for CHGDPRCAPA

The following examples illustrate how to use the **CHGDPRCAPA** command.

CHGDPRCAPA

Example 1

To change the frequency of row insertion to 6 000 seconds (100 minutes) by the Capture program into the IBMSNAP\_CAPMON table:

```
CHGDPRCAPA CAPCTLLIB(ASN) MONITV(6000)
```

This frequency value is stored in the IBMSNAP\_CAPPARMS table that is located in the default ASN library.

Example 2

To change the retention limit, lag limit, trace limit, and monitor limit in the IBMSNAP\_CAPPARMS table located in a Capture control library called LIB1:

```
CHGDPRCAPA CAPCTLLIB(LIB1) RETAIN(6000) LAG(3000) TRCLMT(3000) MONLMT(6000)
```

Example 3

To change the commit interval, which indicates how frequently the Capture program writes changes to the CD and UOW tables:

```
CHGDPRCAPA CAPCTLLIB(ASN) FRCFRQ(360)
```

Related tasks:

- Chapter 9, “Operating the Capture program” on page 117

CRTDPRTBL: Creating the replication control tables (OS/400)

If your replication control tables are accidentally deleted or corrupted, use the Create DPR Tables (**CRTDPRTBL**) command to create them manually.

**Important:** The **CRTDPRTBL** command is the only command that you should use to create OS/400 control tables. Do not use the Replication Center to create the control tables.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

To create replication control tables using the CRTDPRTBL command:

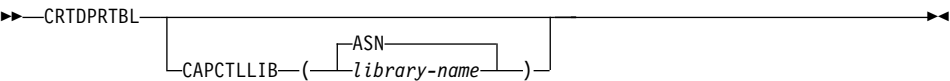


Table 35. CRTDPRTBL command parameter definitions for OS/400

Parameter	Definition and prompts
<b>CAPCTLLIB</b>	Specifies the Capture schema, which is the name of the library where the newly created Capture control tables are placed.  <b>ASN</b> (default) The Capture control tables are placed in the ASN library.  <i>library-name</i> The name of the library where the Capture control tables are placed.

## Examples for CRTDPRTBL

The following examples illustrate how to use the **CRTDPRTBL** command.

### Example 1

To create new replication control tables in the default ASN library:

```
CRTDPRTBL CAPCTLLIB(ASN)
```

### Example 2

To create new replication control tables for a Capture schema called DPRSALES:

```
CRTDPRTBL CAPCTLLIB(DPRSALES)
```

### Related tasks:

- Chapter 2, “Setting up for replication” on page 15

## ENDDPRAPY: Stopping Apply (OS/400)

Use the End DPR Apply (**ENDDPRAPY**) command to stop an Apply program on your local system.

You should stop the Apply program before any planned system down time. You might also want to end the Apply program during periods of peak system activity.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

**To stop an Apply program using the ENDDPRAPY command:**

ENDDPRAPY

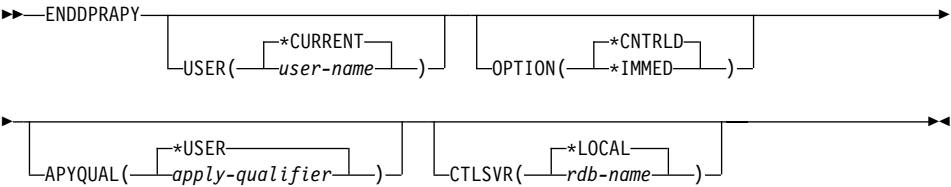


Table 36. ENDDPRAPY command parameter definitions for OS/400

Parameter	Definition and prompts
USER	<p>This parameter is ignored unless the <b>APYQUAL</b> parameter has a value of <b>*USER</b>, in which case this parameter specifies the Apply qualifier associated with the Apply program.</p> <p><b>*CURRENT</b> (default) The Apply program of the user associated with the current job.</p> <p><i>user-name</i> The Apply program of the specified user.</p> <p>When prompting on the <b>ENDDPRAPY</b> command, you can press the F4 key to see a list of users who defined subscriptions.</p>
OPTION	<p>Specifies how to stop the Apply program.</p> <p><b>*CNTRLD</b> (default) The Apply program completes all of its tasks before stopping. These tasks might take a considerable period of time if the Apply program is completing a subscription set.</p> <p><b>*IMMED</b> The Apply program completes all of its tasks with the <b>ENDJOB</b> <b>OPTION(*IMMED)</b> command. The tasks end immediately, without any cleanup. Use this option only after a controlled end is unsuccessful, because it can cause undesirable results. (Unless the Apply program was asleep when you issued the <b>ENDDPRAPY</b> command, you should verify the target table contents.)</p> <p>If the Apply program was performing a full refresh to the target table, the target table might be empty as a result of ending the Apply program before the table was refreshed with the source table contents. If the target table is empty, you must force a full refresh for this replication target.</p> <p>You might find that a subscription set is considered IN USE (the STATUS column in the subscription sets (IBMSNAP_SUBS_SET) table has a value of 1). If it is, reset the value to 0 or -1. This allows the subscription set to be run again by the Apply program.</p>

Table 36. ENDDPRAPY command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>APYQUAL</b>	<p>Specifies the Apply qualifier that is used by the Apply program.</p> <p><b>*USER</b> (default) The user name specified on the <b>USER</b> parameter is the Apply qualifier.</p> <p><i>apply-qualifier</i> The name used to group the subscription sets that this Apply program runs. You can specify a maximum of 18 characters for the Apply qualifier name. This name follows the same naming conventions as a relational database name. You identify the subscriptions being run by the records in the subscription sets (IBMSNAP_SUBS_SET) table with this value in the APPLY_QUAL column.</p> <p>When prompting on the <b>ENDDPRAPY</b> command, you can press the F4 key to see a list of Apply qualifier names with existing subscriptions.</p>
<b>CTLSVR</b>	<p>Specifies the relational database name of the system that contains the Apply control tables.</p> <p><b>*LOCAL</b> (default) The Apply control tables reside locally (from the machine on which you are running the <b>ENDDPRAPY</b> command).</p> <p><i>rdb-name</i> The name of the relational database where the Apply control tables reside. You can use the Work with RDB Directory Entries (<b>WRKRDBDIRE</b>) command to find this name.</p> <p>When prompting on the <b>ENDDPRAPY</b> command, you can press the F4 key to choose from the list of databases in the RDB directory.</p>

## Usage notes

The **ENDDPRAPY** command uses the value of the **APYQUAL** and **CTLSVR** parameters to search the Apply job (IBMSNAP\_APPLY\_JOB) table for the job name, job number, and job user for the referenced Apply program, and ends that job.

**ENDDPRAPY** issues an error message if the following conditions occur:

- If the IBMSNAP\_APPLY\_JOB table does not exist or is corrupted.
- If there is no record in the IBMSNAP\_APPLY\_JOB table for the Apply qualifier and control server name.
- If the Apply job already ended.
- If the user ID running the command is not authorized to end the Apply job.

# ENDDPRAPY

## Examples for ENDDPRAPY

The following examples illustrate how to use the **ENDDPRAPY** command.

### Example 1

To end the Apply program that uses the AQHR Apply qualifier:

```
ENDDPRAPY OPTION(*CNTRLD) APYQUAL(AQHR)
```

The Apply program ends after all of its tasks are completed.

### Example 2

To end the Apply program immediately:

```
ENDDPRAPY OPTION(*IMMED) APYQUAL(AQHR)
```

The tasks of the Apply program end immediately, without any cleanup.

### Example 3

To end an Apply program that uses Apply control tables that reside on a relational database named DB1X:

```
ENDDPRAPY OPTION(*CNTRLD) APYQUAL(AQHR) CTLSVR(DB1X)
```

### Related tasks:

- Chapter 10, “Operating the Apply program” on page 139

---

## ENDDPRCAP: Stopping Capture (OS/400)

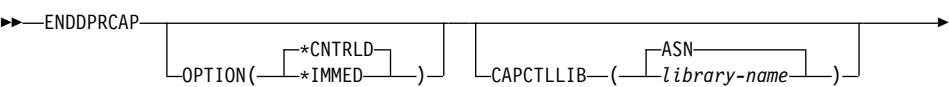
Use the End DPR Capture (**ENDDPRCAP**) command to stop the Capture program.

Use this command to stop the Capture program before shutting down the system. You might also want to stop the program during periods of peak system use to increase the performance of other programs that run on the system.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

### *To stop the Capture program using the ENDDPRCAP command:*



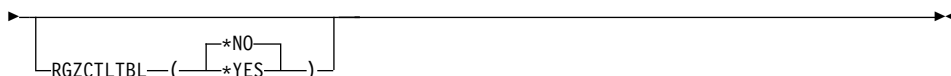


Table 37. ENDDPRCAP command parameter definitions for OS/400

Parameter	Definition and prompts
<b>OPTION</b>	<p>Specifies how to stop the Capture program.</p> <p><b>*CNTRLD</b> (default) The Capture program stops normally after completing all tasks.</p> <p>The <b>ENDDPRCAP</b> command might take longer when you specify the <b>*CNTRLD</b> option because the Capture program completes all of its subordinate processes before stopping.</p> <p><b>*IMMED</b> The Capture program stops normally after completing all tasks with the <b>ENDJOB OPTION(*IMMED)</b> command.</p>
<b>CAPCTLLIB</b>	<p>Specifies the Capture schema, which is the name of the library in which the Capture control tables are located. This library includes the register (IBMSNAP_REGISTER) table, which stores the registration information of the source tables.</p> <p><b>ASN</b> (default) The Capture control tables are in the ASN library. The ASN library is the default library.</p> <p><i>library-name</i> The name of a library that contains the Capture control tables.</p>
<b>RGZCTLTLBL</b>	<p>Specifies whether a Reorganize Physical File Member (<b>RGZPFM</b>) command is performed on the control tables (including the change-data (CD) and unit-of-work (UOW) tables) when the Capture program ends. The system does not recover disk space unless the <b>RGZPFM</b> command process is performed on the tables. The <b>RGZPFM</b> command will not be performed if the control tables are being accessed by an Apply program or by other application programs.</p> <p><b>*NO</b> (default) The <b>RGZPFM</b> command is not performed.</p> <p><b>*YES</b> The <b>RGZPFM</b> command is performed.</p>

## Usage notes

If you use the **ENDJOB** command, temporary objects might be left in the QDP4 library. These objects have the types **\*DTAQ** and **\*USRSPC**, and are named **QDP4nnnnnn**, where **nnnnnn** is the job number of the job that used them. You can delete these objects when the job that used them (identified by the job number in the object name) is not active.

## ENDDPRCAP

If the job under the Capture control library will not end after issuing this command, use the **ENDJOB** command with **\*IMMED** option to end this job and all the journal jobs running in the DB2 DataPropagator for iSeries subsystem. Do not end Apply jobs running in the same subsystem if you want to end only the Capture program.

In rare cases when the Capture control job ends abnormally, the journal jobs created by Capture control job (which is named according to the **CAPCTLLIB** parameter) might still be left running. The only way to end these jobs is to use the **ENDJOB** command with either the **\*IMMED** or **\*CNTRLD** option.

### Examples for ENDDPRCAP

The following examples illustrate how to use the **ENDDPRCAP** command.

#### Example 1

To end the Capture program, which uses Capture control tables in the ASN library, after all processing tasks are completed:

```
ENDDPRCAP OPTION(*CNTRLD) CAPCTLLIB(ASN) RGZCTLTBL(*NO)
```

#### Example 2

To end the Capture program immediately for the Capture schema BSN:

```
ENDDPRCAP OPTION(*IMMED) CAPCTLLIB(BSN) RGZCTLTBL(*NO)
```

#### Example 3

To end the Capture program after all processing tasks are completed and to reorganize the Capture control tables:

```
ENDDPRCAP OPTION(*CNTRLD) CAPCTLLIB(ASN) RGZCTLTBL(*YES)
```

#### Related tasks:

- Chapter 9, “Operating the Capture program” on page 117

---

## GRTDPRAUT: Authorizing users (OS/400)

The Grant DPR Authority (**GRTDPRAUT**) command authorizes a list of users to the replication control tables, so that the users can run the Capture and Apply programs. For example, the authority requirements for the user who is running the Capture and Apply programs might differ from the authority requirements for the user who defines replication sources and targets.

You must have **\*ALLOBJ** authority to grant authorities.

After you type the command name on the command line, you can press the F4 key to display the command syntax.



To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

**To authorize users to the replication control tables using the GRTPRAUT command:**

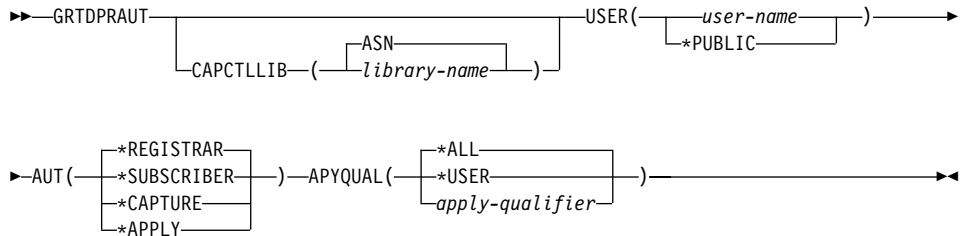


Table 38. GRTPRAUT command parameter definitions for OS/400

Parameter	Definition and prompts
<b>CAPCTLLIB</b>	<p>Specifies the Capture schema, which is the library that contains the replication control tables to which the user is granted authority.</p> <p><b>ASN</b> (default) The Capture control tables reside in the ASN library.</p> <p><i>library-name</i> The name of the library that contains the replication control tables.</p>
<b>USER</b>	<p>Specifies the users who have authority.</p> <p><i>user-name</i> The names of up to 50 users who have authority.</p> <p><b>*PUBLIC</b> Indicates that *PUBLIC authority is granted to the file, but (if insufficient for the task) is used only for those users who have no specific authority, who are not on the authorization list associated with the file, and whose group profile does not have any authority.</p>

Table 38. GRTDPRAUT command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
AUT	<p>Specifies the type of authority being granted.</p> <p><b>*REGISTRAR</b> (default)  The users are granted the authorities to define, change, and remove registrations.</p> <p>For a complete list of authorities with AUT(*REGISTRAR), see Table 39 on page 406.</p> <p><b>*SUBSCRIBER</b>  The users are granted authority to define, change, and remove subscription sets.</p> <p>For a complete list of authorities with AUT(*SUBSCRIBER), see Table 40 on page 407.</p> <p><b>*CAPTURE</b>  The users are granted authority to run the Capture program.</p> <p>For a complete list of authorities granted with AUT(*CAPTURE), see Table 41 on page 407.</p> <p><b>*APPLY</b>  The users are granted authority to run the Apply program.</p> <p>The command does not grant authority to any of the objects that reside on other databases accessed by the Apply program.</p> <p>When an Apply program is invoked, the user associated with the DRDA application server job must also be granted *APPLY authority. If the source is an iSeries server, you should run the <b>GRTDPRAUT</b> command on the source server system, with the application server job user specified on the <b>USER</b> parameter and the Apply qualifier specified on the <b>APYQUAL</b> parameter.</p> <p>Authorities are not granted to the target tables unless the target server is the same as the control server and both reside on the system where the command is run.</p> <p>For a complete list of authorities granted with AUT(*APPLY), see Table 42 on page 409.</p>

Table 38. GRTDPRAUT command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>APYQUAL</b>	<p>Specifies the Apply qualifier to be used by the user as specified with the <b>USER</b> parameter. This parameter is used only when <b>AUT(*APPLY)</b> or <b>AUT(*SUBSCRIBER)</b> is specified.</p> <p><b>*ALL</b> (default) The user is granted authority to run the Apply program or to define and remove subscription sets for <i>all</i> Apply qualifiers.</p> <p><b>*USER</b> The users specified on the <b>USER</b> parameter are granted authority to the subscription sets with an Apply qualifier that is the same as the user name.</p> <p><i>apply-qualifier</i> The user is granted authority to run the Apply program or define and remove subscription sets for the Apply qualifiers associated with this Apply qualifier.</p> <ul style="list-style-type: none"> <li>• The user is granted authority to all replication sources, change-data (CD) tables, and consistent-change data (CCD) tables associated with records in the pruning control (IBMSNAP_PRUNCNTL) table that have a value in the APPLY_QUAL column matching the value input with the <b>APYQUAL</b> parameter.</li> <li>• The user is granted authority to the subscription sets listed in the subscription members (IBMSNAP_SUBS_MEMBR) table that reside on this system.</li> </ul>

## Usage notes

You cannot use the **GRTDPRAUT** command while the Capture or Apply programs are running, or when applications that use the source tables are active because authorizations cannot be changed on files that are in use. The following tables list the authorities granted when you specify:

- **AUT(\*REGISTRAR)**
- **AUT(\*SUBSCRIBER)**
- **AUT(\*CAPTURE)**
- **AUT(\*APPLY)**

on the **GRTDPRAUT** command.

The following table lists the authorities granted when you specify the **AUT(\*REGISTRAR)** parameter on the **GRTDPRAUT** command.

Table 39. Authorities granted with GRTDPRAUT AUT(\*REGISTRAR)

Library	Object	Type	Authorizations
QSYS	capctllib	*LIB	*USE, *ADD
capctllib <sup>1</sup>	QSQJRN	*JRN	*OBJOPR, *OBJMGT
capctllib <sup>1</sup>	QZS8CTLBLK	*USRSPC	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_REGISTER	*FILE	*OBJOPR, *READ, *ADD, *UPDT, *DLT
capctllib <sup>1</sup>	IBMSNAP_REGISTERX	*FILE	*OBJOPR, *READ, *ADD, *UPDT, *DLT
capctllib <sup>1</sup>	IBMSNAP_REGISTERX1	*FILE	*OBJOPR, *READ, *ADD, *UPDT, *DLT
capctllib <sup>1</sup>	IBMSNAP_REGISTERX2	*FILE	*OBJOPR, *READ, *ADD, *UPDT, *DLT
capctllib <sup>1</sup>	IBMSNAP_REG_EXT	*FILE	*OBJOPR, *READ, *ADD, *UPDT, *DLT
capctllib <sup>1</sup>	IBMSNAP_REG_EXTX	*FILE	*OBJOPR, *READ, *ADD, *UPDT, *DLT
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTL	*FILE	*OBJOPR, *READ
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTLX	*FILE	*OBJOPR, *READ
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTLX1	*FILE	*OBJOPR, *READ
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTLX2	*FILE	*OBJOPR, *READ
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTLX3	*FILE	*OBJOPR, *READ
ASN	ASN4B*	*SQLPKG	*USE
ASN	ASN4C*	*SQLPKG	*USE

**Notes:**

1. The entry *capctllib* in the Library column refers to the value passed to the **CAPCTLLIB** parameter of the **GRTDPRAUT** command; this command updates authority to only one Capture control library at a time.

The following table lists the authorities granted when you specify the AUT(\*SUBSCRIBER) parameter on the **GRTDPRAUT** command.

Table 40. Authorities granted with GRDPRAUT AUT(\*SUBSCRIBER)

Library	Object	Type	Authorizations
QSYS	ASN	*LIB	*OBJOPR, *READ, *ADD, *EXECUTE
QSYS	capctllib	*LIB	*OBJOPR, *READ, *ADD, *EXECUTE
ASN	IBMSNAP_SUBS_SET	*FILE	*CHANGE
ASN	IBMSNAP_SUBS_COLS	*FILE	*CHANGE
ASN	IBMSNAP_SUBS_EVENT	*FILE	*CHANGE
ASN	IBMSNAP_SUBS_STMTS	*FILE	*CHANGE
ASN	IBMSNAP_SUBS_MEMBR	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_REGISTER	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REG_EXT	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTL	*FILE	*OBJOPR, *READ, *DLT, *ADD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTLX	*FILE	*USE
ASN	ASN4A*	*SQLPKG	*USE
ASN	ASN4U*	*SQLPKG	*USE

**Notes:**

1. The entry *capctllib* in the Library column refers to the value passed to the **CAPCTLLIB** parameter of the **GRDPRAUT** command; this command updates authority to only one Capture control library at a time.

The following table lists the authorities granted when you specify the AUT(\*CAPTURE) parameter on the **GRDPRAUT** command.

Table 41. Authorities granted with GRDPRAUT AUT(\*CAPTURE)

Library	Object	Type	Authorizations
QSYS	capctllib	*LIB	*OBJOPR, *OBJMGT, *READ, *EXECUTE
QSYS	QDP4	*LIB	*OBJOPR, *ADD, *READ, *EXECUTE
capctllib <sup>1</sup>	QZSN	*MSGQ	*CHANGE

Table 41. Authorities granted with GRTDPRAUT AUT(\*CAPTURE) (continued)

Library	Object	Type	Authorizations
capctllib <sup>1</sup>	IBMSNAP_REGISTER	*FILE	*OBJOPR, *OBJMGT, *READ, *ADD, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REGISTERX	*FILE	*OBJOPR, *OBJMGT, *READ, *ADD, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REGISTERX1	*FILE	*OBJOPR, *OBJMGT, *READ, *ADD, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REGISTERX2	*FILE	*OBJOPR, *OBJMGT, *READ, *ADD, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REG_EXT	*FILE	*OBJOPR, *OBJMGT, *READ, *ADD, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REG_EXTX	*FILE	*OBJOPR, *OBJMGT, *READ, *ADD, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTL	*FILE	*OBJOPR, *OBJMGT, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTLX	*FILE	*OBJOPR, *OBJMGT, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTLX1	*FILE	*OBJOPR, *OBJMGT, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTLX2	*FILE	*OBJOPR, *OBJMGT, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTLX3	*FILE	*OBJOPR, *OBJMGT, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_CAPTRACE	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_CAPTRACEX	*FILE	*CHANGE

Table 41. Authorities granted with GRTDPRAUT AUT(\*CAPTURE) (continued)

Library	Object	Type	Authorizations
capctllib <sup>1</sup>	IBMSNAP_RESTART	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_RESTARTX	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_AUTHTKN	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_AUTHTKNX	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_UOW	*FILE	*OBJOPR, *OBJMGT, *READ, *UPD, *DLT, *ADD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_UOW_IDX	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_PRUNE_SET	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_PRUNE_SETX	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_CAPPARMS	*FILE	*READ, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_SIGNAL	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_SIGNALX	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_CAPMON	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_CAPMONX	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_PRUNE_LOCK	*FILE	*CHANGE
ASN	ASN4B*	*SQLPKG	*USE
ASN	ASN4C*	*SQLPKG	*USE
ASN	QZS8CTLBLK	*USRSPC	*CHANGE

**Notes:**

1. The entry *capctllib* in the Library column refers to the value passed to the **CAPCTLLIB** parameter of the **GRTDPRAUT** command; this command updates authority to only one Capture control library at a time.

The following table lists the authorities granted when you specify the AUT(\*APPLY) parameter on the **GRTDPRAUT** command.

Table 42. Authorities granted with GRTDPRAUT AUT(\*APPLY)

Library	Object	Type	Authorizations
QSYS	ASN	*LIB	*OBJOPR, *READ, *EXECUTE
QSYS	capctllib	*LIB	*OBJOPR, *READ, *EXECUTE

Table 42. Authorities granted with GRTDPRAUT AUT(\*APPLY) (continued)

Library	Object	Type	Authorizations
QDP4	QZSNAPV2	*PGM	*OBJOPR, *READ, *OBMGT, *OBJALTER, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REGISTER	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REGISTERX	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REGISTERX1	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REGISTERX2	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REGISTER_EXT	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_REGISTER_EXTX	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_SIGNAL	*FILE	*OBJOPR, *READ, *UPD, *ADD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_SIGNALX	*FILE	*OBJOPR, *READ, *UPD, *ADD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_PRUNE_LOCK	*FILE	*CHANGE
capctllib <sup>1</sup>	IBMSNAP_UOW	*FILE	*OBJOPR, *READ, *UPD, *ADD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_PRUNCNTL	*FILE	*OBJOPR, *READ, *UPD, *ADD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_AUTHTKN	*FILE	*OBJOPR, *READ, *UPD, *ADD, *EXECUTE
capctllib <sup>1</sup>	IBMSNAP_AUTHTKNX	*FILE	*OBJOPR, *READ, *UPD, *ADD, *EXECUTE
ASN	IBMSNAP_SUBS_SET	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
ASN	IBMSNAP_SUBS_SETX	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE



Table 42. Authorities granted with GRTDPRAUT AUT(\*APPLY) (continued)

Library	Object	Type	Authorizations
ASN	IBMSNAP_APPLYTRAIL	*FILE	*OBJOPR, *READ, *UPD, *ADD, *EXECUTE
ASN	IBMSNAP_APPLYTRACE	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
ASN	IBMSNAP_APPLYTRACX	*FILE	*OBJOPR, *READ, *UPD, *EXECUTE
ASN	IBMSNAP_SUBS_COLS	*FILE	*USE
ASN	IBMSNAP_SUBS_EVENT	*FILE	*USE
ASN	IBMSNAP_SUBS_STMTS	*FILE	*USE
ASN	IBMSNAP_SUBS_MEMBR	*FILE	*USE
ASN	ASN4A*	*SQLPKG	*USE
ASN	ASN4U*	*SQLPKG	*USE
ASN	IBMSNAP_APPLY_JOB	*FILE	*OBJOPR, *READ, *UPD, *ADD, *EXECUTE

**Notes:**

1. The entry *capctllib* in the Library column refers to the value passed to the **CAPCTLLIB** parameter of the **GRTDPRAUT** command; this command updates authority to only one Capture control library at a time.

## Examples for GRTDPRAUT

The following examples illustrate how to use the **GRTDPRAUT** command.

### Example 1

To authorize a user named USER1 to define and modify registrations:

```
GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*REGISTRAR)
```

### Example 2

To authorize a user named USER1 to define and modify subscription sets:

```
GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*SUBSCRIBER)
```

### Example 3

To authorize a user named USER1 to run Capture programs:

```
GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*CAPTURE)
```

### Example 4

To authorize a user named USER1 to define and modify existing subscription sets that are associated with Apply qualifier A1:

## GRTDPRAUT

```
GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*SUBSCRIBER) APYQUAL(A1)
```

### Example 5

To authorize a user to run the Apply program on the control server system for all subscription sets associated with Apply qualifier A1, where the target server is the same as the control server:

1. Run the following command on the system where the Apply program will run:

```
GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*APPLY) APYQUAL(A1)
```

2. Run the appropriate **GRTDPRAUT** command on the source server system:

- If the application server job on the source server used by the Apply program runs under user profile USER1, run the following command on the source server systems:

```
GRTDPRAUT CAPCTLLIB(ASN) USER(USER1) AUT(*APPLY) APYQUAL(A1)
```

- If the application server job on the source server used by the Apply program runs under a different user profile, for example, QUSER, the command is:

```
GRTDPRAUT CAPCTLLIB(ASN) USER(QUSER) AUT(*APPLY) APYQUAL(A1)
```

### Related tasks:

- Chapter 2, “Setting up for replication” on page 15

### Related reference:

- “asnpwd: Maintaining password files (UNIX and Windows)” on page 335
- “RVKDPRAUT: Revoking authority (OS/400)” on page 425

---

## INZDPRCAP: Reinitializing DPR Capture (OS/400)

Use the Initialize DPR Capture (**INZDPRCAP**) command to initialize the Capture program by directing the Capture program to work with an updated list of source tables.

Source tables under the control of a Capture program can change while the Capture program is running. Use the **INZDPRCAP** command to ensure that the Capture program processes the most up-to-date replication sources.

The Capture program must be running before you can run this command.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

**To initialize the Capture program using the INZDPRCAP command:**

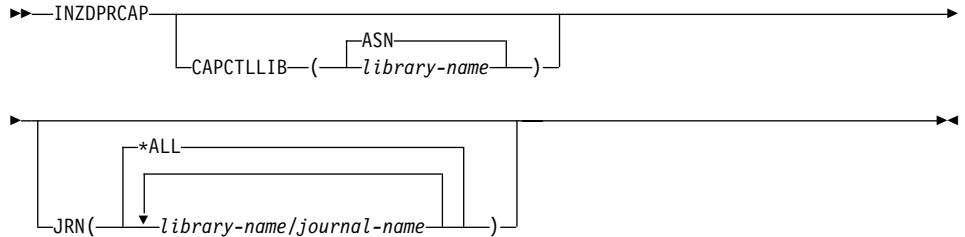


Table 43. INZDPRCAP command parameter definitions for OS/400

Parameter	Definition and prompts
<b>CAPCTLLIB</b>	<p>Specifies the Capture schema, which is the name of the library in which the Capture control tables reside.</p> <p><b>ASN</b> (default) The Capture control tables reside in the ASN library. The ASN library is the default library.</p> <p><i>library-name</i> The name of a library that contains the Capture control tables.</p>
<b>JRN</b>	<p>Specifies a subset of up to 50 journals that you want the Capture program to work with. The Capture program starts processing all the source tables that are currently journaled to this journal.</p> <p><b>*ALL</b> (default) The Capture program works with all the journals.</p> <p><i>library-name/journal-name</i> The qualified name of the journal that you want the Capture program to work with.</p>

## Examples for INZDPRCAP

The following examples illustrate how to use the **INZDPRCAP** command.

### Example 1

To initialize a Capture program using the QSQJRN journal under a library named TRAINING:

```
INZDPRCAP CAPCTLLIB(ASN) JRN(TRAINING/QSQJRN)
```

The Capture control tables reside in the default ASN schema.

## INZDPRCAP

### Example 2

To initialize a Capture program that works with all the journals:

```
INZDPRCAP CAPCTLLIB(BSN) JRN(*ALL)
```

The Capture control tables reside in a schema called BSN.

### Related tasks:

- Chapter 9, “Operating the Capture program” on page 117

---

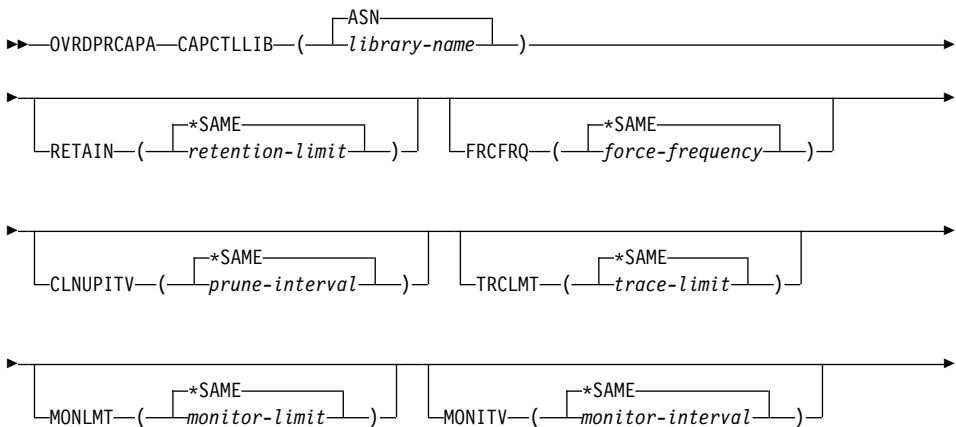
## OVRDPRCAPA: Overriding DPR capture attributes (OS/400)

Use the Override DPR Capture attributes (**OVRDPRCAPA**) command to alter the behavior of a running Capture program. This command alters the program behavior by overriding the values that were passed to the Capture program from the Capture parameters (IBMSNAP\_CAPPARMS) table or from the **STRDPRCAP** command when the Capture program started.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

***To override the attributes of a Capture program using the OVRDPRCAPA command:***



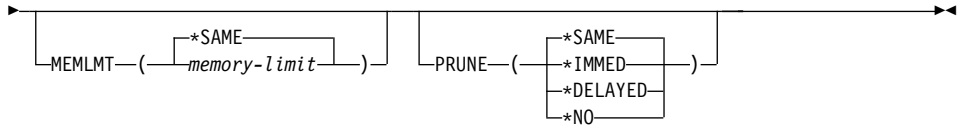


Table 44. OVRDPRCAPA command parameter definitions for OS/400

Parameter	Definition and prompts
<b>CAPCTLLIB</b>	<p>Specifies the Capture schema, which is the name of the library in which the Capture control tables reside. This library includes the register (IBMSNAP_REGISTER) table, which stores the registration information of the source tables. This parameter is required.</p> <p><b>ASN</b> (default) The Capture control tables reside in the ASN library.</p> <p><i>library-name</i> The name of a library that contains the Capture control tables. You can create this library using the <b>CRTPRTBL</b> command with the <b>CAPCTLLIB</b> parameter.</p>
<b>RETAIN</b>	<p>Specifies the number of minutes that data is retained in the change-data (CD), unit-of-work (UOW), signal (IBMSNAP_SIGNAL), and Apply qualifier cross-reference (IBMSNAP_AUTHTKN) tables before the data is removed.</p> <p>This value works with the <b>CLNUPITV</b> parameter value from the Start DPR Capture (<b>STRDPRCAP</b>) command. First, the Capture program deletes any CD, UOW, IBMSNAP_SIGNAL, or IBMSNAP_AUTHTKN rows that are older than the oldest currently running Apply program. Then, a new or remaining row from the CD, UOW, IBMSNAP_SIGNAL, or IBMSNAP_AUTHTKN table is subsequently deleted when its age reaches the value of the <b>RETAIN</b> parameter.</p> <p>Ensure that the Apply intervals are set to copy changed information before the data reaches this <b>RETAIN</b> parameter value to prevent inconsistent data in your tables. If the data becomes inconsistent, the Apply program performs a full refresh.</p> <p>The default is 10 080 minutes (seven days). The maximum is 35000000 minutes.</p> <p><b>*SAME</b> (default) This value is not changed.</p> <p><i>retention-limit</i> The new retention limit value.</p>

Table 44. OVRDPRCAPA command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>FRCFRQ</b>	<p>Specifies how often (from 30 to 600 seconds) the Capture program writes changes to the change-data (CD) and unit-of-work (UOW) tables.</p> <p>The Capture program makes these changes available to the Apply program either when the buffers are filled or when the <b>FRCFRQ</b> time limit expires, whichever is sooner. This parameter value affects the amount of time that it takes for the Capture program to respond to changes from the Initialize DPR Capture (<b>INZDPRCAP</b>) command.</p> <p>Use this parameter to make changes more readily available to the Apply program on servers with a low rate of source table changes. The <b>FRCFRQ</b> parameter value is a global value used for all registered source tables. Setting the <b>FRCFRQ</b> value to a low number can affect system performance.</p> <p>The default is 30 seconds.</p> <p><b>*SAME</b> (default) This value is not changed.</p> <p><i>force-frequency</i> The new number of seconds that the Capture program keeps CD and UOW table changes in buffer space before making these changes available to the Apply program.</p>

Table 44. OVRDPRCAPA command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>CLNUPITV</b>	<p>Specifies the maximum amount of time (in hours) before the Capture program prunes old records from the change-data (CD), unit-of-work (UOW), signal (IBMSNAP_SIGNAL), Capture monitor (IBMSNAP_CAPMON), Capture trace (IBMSNAP_CAPTRACE), and Apply qualifier cross-reference (IBMSNAP_AUTHTKN) tables.</p> <p>This parameter works with the <b>RETAIN</b> parameter to control pruning of the CD, UOW, IBMSNAP_SIGNAL, and IBMSNAP_AUTHTKN tables, with the <b>MONLMT</b> parameter to control pruning of the IBMSNAP_CAPMON table, and with the <b>TRCLMT</b> parameter to control pruning of the IBMSNAP_CAPTRACE table.</p> <p>(Use the <b>STRDPRCAP</b> command to set the <b>RETAIN</b>, <b>MONLMT</b>, and <b>TRCLMT</b> parameters for a Capture program.)</p> <p>The value of the <b>CLNUPITV</b> parameter is automatically converted from hours to seconds and is stored in the PRUNE_INTERVAL column of the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p><b>*SAME</b> (default) This Capture attribute value is not changed.</p> <p><i>prune-interval</i> The pruning interval expressed as a specific number of hours (1 to 100).</p>
<b>TRCLMT</b>	<p>Specifies the trace limit, which indicates how frequently the Capture trace (IBMSNAP_CAPTRACE) table is pruned.</p> <p><b>*SAME</b> (default) The Capture program continues using the current trace limit value.</p> <p><i>trace-limit</i> The number of minutes between each pruning operation of the IBMSNAP_CAPTRACE table.</p>
<b>MONLMT</b>	<p>Specifies the monitor limit, which indicates how frequently the Capture monitor (IBMSNAP_CAPMON) table is pruned.</p> <p><b>*SAME</b> (default) The Capture program continues using the current monitor limit value.</p> <p><i>monitor-limit</i> The number of minutes between each pruning operation of the IBMSNAP_CAPMON table.</p>

Table 44. OVRDPRCAPA command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
MONITV	<p>Specifies the monitor interval (in seconds), which indicates how frequently the Capture program inserts rows into the Capture monitor (IBMSNAP_CAPMON) table.</p> <p><b>*SAME</b> (default) The Capture program continues using the current monitor interval value.</p> <p><i>monitor-interval</i> The number of seconds between row insertion into the IBMSNAP_CAPMON table. The monitor interval must be at least 120 seconds (two minutes). If you type a number that is less than 120, the command automatically sets this parameter value to 120.</p>
MEMLMT	<p>Specifies the maximum size (in megabytes) of memory that the Capture journal job can use.</p> <p><b>*SAME</b> (default) The Capture program continues using the current memory limit value.</p> <p><i>memory-limit</i> The maximum number of megabytes for memory.</p>



Table 44. OVRDPRCAPA command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>PRUNE</b>	<p>Use this parameter to change the way that the Capture program prunes rows from the change-data (CD), unit-of-work (UOW), signal (IBMSNAP_SIGNAL), Capture monitor (IBMSNAP_CAPMON), Capture trace (IBMSNAP_CAPTRACE), and Apply qualifier cross-reference (IBMSNAP_AUTHTKN) tables.</p> <p><b>*SAME</b> (default) The Capture program continues using the pruning parameters that you specified when you started the <b>STRDPRCAP</b> command.</p> <p><b>*IMMED</b> The Capture program starts pruning the tables immediately, regardless of the value of the <b>CLNUPITV</b> parameter that you specified when you started the <b>STRDPRCAP</b> command.</p> <p><b>*DELAYED</b> The Capture program prunes the old rows at the end of the specified pruning interval.</p> <p>PRUNE(*DELAYED) does not affect the frequency of pruning if you set the second part of the <b>CLNUPITV</b> parameter to <b>*IMMED</b> or <b>*DELAYED</b> on the <b>STRDPRCAP</b> command. However, PRUNE(*DELAYED) <i>does</i> initiate pruning if you set the second part of the <b>CLNUPITV</b> parameter to <b>*NO</b> when you started the <b>STRDPRCAP</b> command.</p> <p><b>*NO</b> The Capture program does not initiate pruning. This value overrides the <b>CLNUPITV</b> parameter setting from the <b>STRDPRCAP</b> command.</p>

## Examples for OVRDPRCAPA

The following examples illustrate how to use the **OVRDPRCAPA** command.

### Example 1

To change the pruning parameters of the CD, UOW, IBMSNAP\_SIGNAL, IBMSNAP\_CAPMON, IBMSNAP\_CAPTRACE, and IBMSNAP\_AUTHTKN tables (which reside under the default ASN library) and to change the IBMSNAP\_CAPMON monitor interval and memory limit of Capture journal jobs in a running Capture program:

```
OVRDPRCAPA CAPCTLLIB(ASN) CLNUPITV(12) MONITV(600) MEMLM(64)
```

### Example 2

To initiate pruning of the CD, UOW, IBMSNAP\_SIGNAL, IBMSNAP\_CAPMON, IBMSNAP\_CAPTRACE, and IBMSNAP\_AUTHTKN tables, which reside in the BSN library:

```
OVRDPRCAPA CAPCTLLIB(BSN) PRUNE(*IMMED)
```

**Related tasks:**

- Chapter 9, “Operating the Capture program” on page 117

**Related reference:**

- “asnccmd: Operating Capture (UNIX, Windows, z/OS)” on page 322

**RMVDPRREG: Removing a DPR registration (OS/400)**

Use the Remove DPR registration (**RMVDPRREG**) command to remove a single source table from the register (IBMSNAP\_REGISTER) table so that the source table is no longer used for data propagation.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

**To remove a DPR registration using the RMVDPRREG command:**

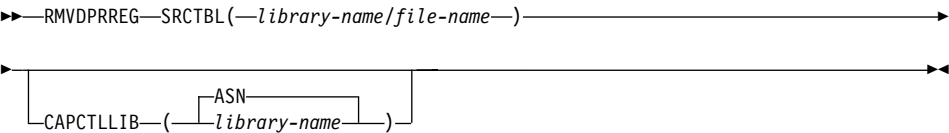


Table 45. RMVDPRREG command parameter definitions for OS/400

Parameter	Definition and prompts
SRCTBL	Identifies the registration that you want to remove. This is a required parameter.  <i>library-name/file-name</i> The qualified name of the registered file.
CAPCTLLIB	Specifies the Capture schema, which is the name of the library in which the Capture control tables reside.  ASN (default) The Capture control tables are in the ASN library.  <i>library-name</i> The name of a library containing the Capture control tables.

## Examples for RMVDPRREG

The following examples illustrate how to use the **RMVDPRREG** command.

### Example 1

To remove the registration for the source table named **EMPLOYEE** of the **HR** library in the default **ASN Capture** schema:

```
RMVDPRREG SRCTBL(HR/EMPLOYEE)
```

### Example 2

To remove the registration for the source table named **SALES** of the **DEPT** library under a **Capture** schema called **BSN**:

```
RMVDPRREG SRCTBL(DEPT/SALES) CAPCTLLIB(BSN)
```

### Related tasks:

- Chapter 12, “Making changes to your replication environment” on page 183

---

## RMVDPRSUB: Removing a DPR subscription set (OS/400)

Use the Remove DPR subscription set (**RMVDPRSUB**) command to remove a subscription set. If you set the **RMVMBRS** parameter to **\*YES**, this command removes the subscription set and all of its members.

After you type the command name on the command line, you can press the **F4** key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the **F1** key. To display a description of a specific parameter, place the cursor on that parameter and press the **F1** key.

### To remove a subscription set using the **RMVDPRSUB** command:

```

▶▶—RMVDPRSUB—APYQUAL—(—apply-qualifier—)—SETNAME—(—set-name—)—▶▶
|
|  CTLSVR—(—☐*LOCAL—☐—)  RMVREG—(—☐*NO—☐*YES—)
|  rdn-name
|
|  DLTTGTBL—(—☐*NO—☐*YES—)  RMVMBRS—(—☐*NO—☐*YES—)
|  rdn-name
|
▶▶

```

Table 46. RMVDPRSUB command parameter definitions for OS/400

Parameter	Definition and prompts
<b>APYQUAL</b>	Specifies the Apply qualifier that the Apply program uses to identify the subscription set. This parameter is required.  <i>apply-qualifier</i> The name of the Apply qualifier.
<b>SETNAME</b>	Specifies the name of the subscription set. This parameter is required.  <i>set-name</i> The name of the subscription set. You receive an error message if you enter a subscription-set name that does not exist for the specified Apply qualifier.
<b>CTLSVR</b>	Specifies the relational database name of the system that contains the Apply control tables.  <b>*LOCAL</b> (default) The Apply control tables reside locally (on the machine from which you are running the <b>RMVDPRSUB</b> command).  <i>rdb-name</i> The name of the relational database where the Apply control tables reside. You can use the Work with RDB Directory Entries ( <b>WRKRDBDIRE</b> ) command to find this name.
<b>RMVREG</b>	Specifies whether this command removes the registrations that are associated with the target tables of all the subscription-set members in the subscription set. Use this parameter only if you have set the <b>RMVMBRS</b> parameter to <b>*YES</b> .  <b>*NO</b> (default) The registrations are not removed.  <b>*YES</b> The registrations are removed.
<b>DLTTGTTBL</b>	Specifies whether this command drops the target tables of the subscription-set members after the subscription set is removed. Use this parameter only if you set the <b>RMVMBRS</b> parameter to <b>*YES</b> .  <b>*NO</b> (default) The target tables are not dropped.  <b>*YES</b> The target tables are dropped.

Table 46. RMVDPRSUB command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>RMVMBRS</b>	Specifies whether this command removes the subscription set and all the members in that subscription set.  <b>*NO</b> (default) The subscription set is not removed if there are existing members in the subscription set.  <b>*YES</b> The subscription set and all its subscription-set members are removed.

## Examples for RMVDPRSUB

The following examples illustrate how to use the **RMVDPRSUB** command.

### Example 1

To remove a subscription set named SETHR that contains no subscription-set members:

```
RMVDPRSUB APYQUAL(AQHR) SETNAME(SETHR)
```

### Example 2

To remove a subscription set named SETHR and all its subscription-set members:

```
RMVDPRSUB APYQUAL(AQHR) SETNAME(SETHR) RMVMBRS(*YES)
```

### Example 3

To remove a subscription set named SETHR, all its subscription-set members, and the associated registrations:

```
RMVDPRSUB APYQUAL(AQHR) SETNAME(SETHR) RMVREG(*YES) RMVMBRS(*YES)
```

### Related tasks:

- Chapter 12, “Making changes to your replication environment” on page 183
- Chapter 4, “Subscribing to sources” on page 63

---

## RMVDPRSUBM: Removing a DPR subscription-set member (OS/400)

Use the Remove DPR subscription-set member (**RMVDPRSUBM**) command to remove a single subscription-set member from a subscription set.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

# RMVDPRSUBM

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

**To remove a single subscription-set member from a subscription set using the RMVDPRSUBM command:**

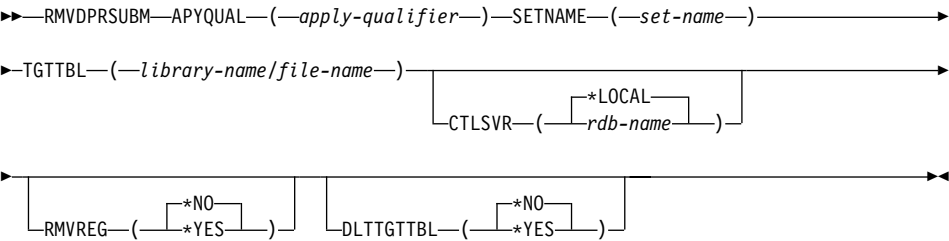


Table 47. RMVDPRSUBM command parameter definitions for OS/400

Parameter	Definition and prompts
APYQUAL	<p>Specifies the Apply qualifier that the Apply program uses to identify the subscription set. This parameter is required.</p> <p><i>apply-qualifier</i></p> <p>The name of the Apply qualifier.</p>
SETNAME	<p>Specifies the name of the subscription set. This parameter is required.</p> <p><i>set-name</i></p> <p>The name of the subscription set. You receive an error message if you enter a subscription-set name that does not exist for the specified Apply qualifier.</p>
TGTTBL	<p>Specifies the target table that is registered for the subscription-set member. This parameter is required.</p> <p><i>library-name/file-name</i></p> <p>The qualified name of the target table.</p>
CTLSVR	<p>Specifies the relational database name of the system that contains the Apply control tables.</p> <p><b>*LOCAL</b> (default)</p> <p>The Apply control tables reside locally (on the machine from which you are running the <b>RMVDPRSUBM</b> command).</p> <p><i>rdb-name</i></p> <p>The name of the relational database where the Apply control tables reside. You can use the Work with RDB Directory Entries (<b>WRKRDBDIRE</b>) command to find this name.</p>

Table 47. RMVDPRSUBM command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>RMVREG</b>	Specifies whether this command removes the registration that is associated with the target table for the subscription-set member.  *NO (default) The registration is not removed.  *YES The registration is removed.
<b>DLTTGTTBL</b>	Specifies whether this command drops the target table of the subscription-set member after the subscription-set member is removed.  *NO (default) The target table is not dropped.  *YES The target table is dropped.

## Examples for RMVDPRSUBM

The following examples illustrate how to use the **RMVDPRSUBM** command.

### Example 1

To remove a subscription-set member, which uses a target table named EMP, from the SETEMP subscription set on the relational database named RMTRDB1:

```
RMVDPRSUBM APYQUAL(AQHR) SETNAME(SETEMP) TGTTBL(TGTEMP/EMP) CTLSVR(RMTRDB1)
```

### Example 2

To remove a subscription-set member from the SETHR subscription set, remove the registration, and then drop the table:

```
RMVDPRSUBM APYQUAL(AQHR) SETNAME(SETHR) TGTTBL(TGTHR/YTD TAX) RMVREG(*YES)
DLTTGTTBL(*YES)
```

### Related tasks:

- Chapter 4, “Subscribing to sources” on page 63

## RVKDPRAUT: Revoking authority (OS/400)

The Revoke DPR Authority (**RVKDPRAUT**) command revokes authority to the replication control tables so that users can no longer define or modify replication sources and subscription sets.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

# RVKDPRAUT

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

**To revoke authority to the replication control tables using the RVKDPRAUT command:**

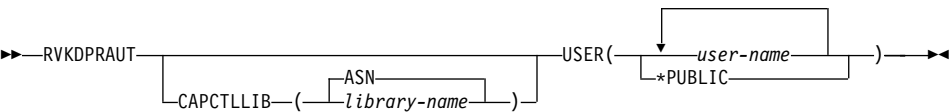


Table 48. RVKDPRAUT command parameter definitions for OS/400

Parameter	Definition and prompts
<b>CAPCTLLIB</b>	<p>Specifies the Capture schema, which is the name of the library under which user authority is being revoked.</p> <p><b>ASN (default)</b> The Capture control tables reside in the ASN library.</p> <p><i>library-name</i> The name of the library that contains the replication control tables.</p>
<b>USER</b>	<p>Specifies the users whose authority is revoked. This parameter is required.</p> <p><i>user-name</i> Specifies the names of up to 50 users whose authority is revoked.</p> <p><b>*PUBLIC</b> Specifies that authority is revoked from all users without specific authority, who are not on the authorization list, and whose group profile does not have any authority.</p>

## Usage notes

The command returns an error message if any of the following conditions exist:

- A specified user does not exist.
- The user running the command is not authorized to the specified user profiles.
- The user running the command does not have permission to revoke authorities to the DB2 DataPropagator for iSeries control tables.
- The DB2 DataPropagator for iSeries control tables do not exist.
- The Capture or Apply programs are running.





## STRDPRAPY

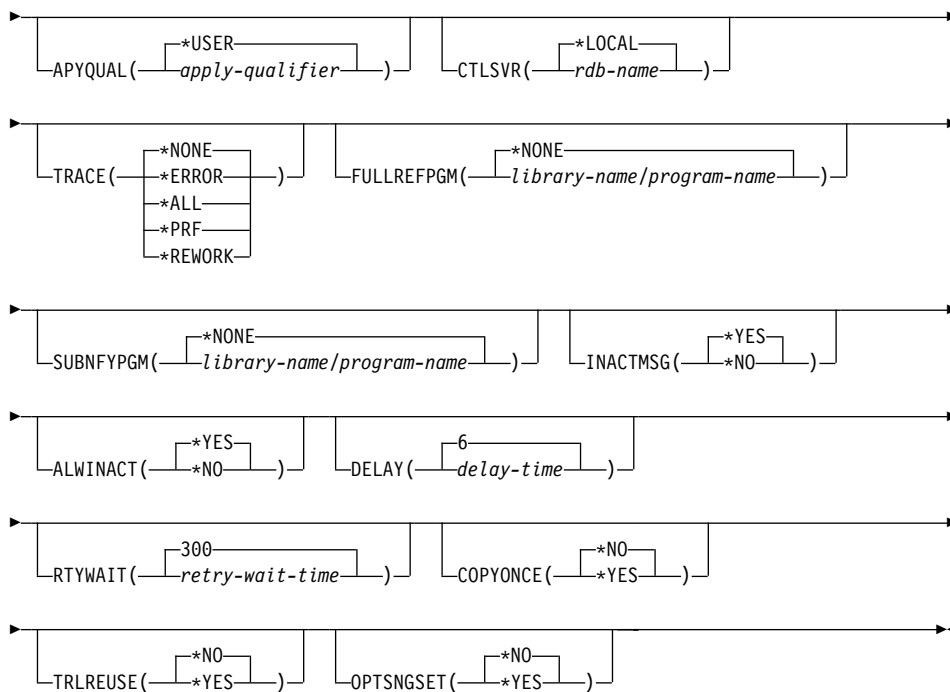


Table 49. STRDPRAPY command parameter definitions for OS/400

Parameter	Definition and prompts
<b>USER</b>	<p>Specifies the name of the user ID for which to start the Apply program. When you run this command, you must be authorized (have *USE rights) to the specified user profile; the Apply program runs under this specified user profile.</p> <p>The control tables are located on the relational database specified by the <b>CTLSVR</b> parameter. The same control tables are used regardless of the value specified on the <b>USER</b> parameter.</p> <p><b>*CURRENT</b> (default) The user ID associated with the current job is the same user ID associated with this Apply program.</p> <p><b>*JOBID</b> The user ID specified in the job description associated with this Apply program. The job description cannot specify USER(*RQD).</p> <p><i>user-name</i> The user ID associated with this Apply program. The following IBM-supplied objects are <i>not</i> valid on this parameter: QDBSHR, QDFTOWN, QDOC, QLPAUTO, QLPINSTALL, QRJE, QSECOFR, QSPL, QSYS, or QTSTRQS.</p> <p>When prompting on the <b>STRDPRAPY</b> command, you can press the F4 key to see a list of users who defined subscription sets.</p>
<b>JOBID</b>	<p>Specifies the name of the job description to use when submitting the Apply program.</p> <p><b>*LIBL/QZSNDPR</b> (default) The default job description provided with DB2 DataPropagator for iSeries.</p> <p><i>library-name/job-description-name</i> The name of the job description used for the Apply program.</p>

Table 49. STRDPRAPY command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
APYQUAL	<p>Specifies the Apply qualifier to be used by the Apply program. All subscriptions sets that are grouped together with this Apply qualifier are run by the Apply program.</p> <p><b>*USER</b> (default) The <b>USER</b> parameter value that you enter is used as the name of the Apply qualifier.</p> <p><i>apply-qualifier</i> The name used to group the subscription sets that are to be run by this Apply program. You can specify a maximum of 18 characters for the Apply qualifier name. This name follows the same naming conventions as a relational database name.</p> <p>When prompting on the <b>STRDPRAPY</b> command, you can press the F4 key to see a list of Apply qualifier names with existing subscription sets.</p>
CTLSVR	<p>Specifies the relational database name of the system that contains the Apply control tables.</p> <p><b>*LOCAL</b> (default) The Apply control tables reside locally (on the machine where you are running the <b>STRDPRAPY</b> command).</p> <p><i>rdb-name</i> The name of the relational database where the Apply control tables reside. You can use the Work with RDB Directory Entries (<b>WRKRDBDIRE</b>) command to find this name.</p> <p>When prompting on the <b>STRDPRAPY</b> command, you can press the F4 key to see a list of available RDB names.</p>

Table 49. STRDPRAPY command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>TRACE</b>	<p>Specifies whether the Apply program should generate a trace. The Apply program writes the trace data to a spool file called QPZSNATRC.</p> <p><b>*NONE</b> (default) No trace is generated.</p> <p><b>*ERROR</b> The trace contains error information only.</p> <p><b>*ALL</b> The trace contains error and execution flow information.</p> <p><b>*PRF</b> The trace contains information that can be used to analyze performance at different stages of the Apply program execution.</p> <p><b>*REWORK</b> The trace contains information about rows that were reworked by the Apply program.</p>
<b>FULLREFPGM</b>	<p>Specifies whether the Apply program is to invoke an exit routine to initialize a target table. When the Apply program is ready to perform a full refresh of a target table, the Apply program invokes the specified exit routine rather than performing the full refresh itself.</p> <p>When a full-refresh exit routine is used by the Apply program, the value of the ASNLOAD column in the Apply trail (IBMSNAP_APPLYTRAIL) table is Y.</p> <p>For examples and more information, see “Refreshing target tables using the ASNLOAD exit routine” on page 154.</p> <p><b>*NONE</b> (default) A full-refresh exit routine is not used.</p> <p><i>library-name/program-name</i> The qualified name of the program that is called by the Apply program performing a full refresh of a target table. For example, to call program ASNLOAD in library DATAPROP, the qualified name is DATAPROP/ASNLOAD.</p>

Table 49. STRDPRAPY command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>SUBNFYPGM</b>	<p>Specifies whether the Apply program is to invoke an exit routine when the program finishes processing a subscription set. Input to the exit routine includes the subscription set name, Apply qualifier, completion status, and statistics with the number of rejects.</p> <p>The notify program allows you to examine the unit-of-work (UOW) table to determine when transactions have been rejected and when to take further actions, such as issuing a message or generating an event.</p> <p>For more information, see “Modifying the ASNDONE exit routine (OS/400)” on page 152.</p> <p><b>*NONE</b> (default) An exit routine is not used.</p> <p><i>library-name/program-name</i> The qualified name of the exit routine program called by the Apply program when processing a subscription set. For example, to call program APPLYDONE in library DATAPROP, the qualified name is DATAPROP/APPLYDONE.</p>
<b>INACTMSG</b>	<p>Specifies whether the Apply program is to generate a message whenever the program completes its work and becomes inactive for a period of time.</p> <p><b>*YES</b> (default) The Apply program generates message ASN1044 before beginning a period of inactivity. Message ASN1044 indicates how long the Apply program remains inactive.</p> <p><b>*NO</b> No message is generated.</p>
<b>ALWINACT</b>	<p>Specifies whether the Apply program can run in an inactive (sleep) state.</p> <p><b>*YES</b> (default) The Apply program sleeps if there is nothing to process.</p> <p><b>*NO</b> If the Apply program has nothing to process, the job that submitted and started the Apply program ends.</p>

Table 49. STRDPRAPY command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>DELAY</b>	<p>Specifies the delay time (in seconds) at the end of each Apply program cycle when continuous replication is used.</p> <p><b>6</b> (default) The delay time is six seconds.</p> <p><i>delay-time</i> The delay time, entered as a number between 0 and 6 inclusive.</p>
<b>RTYWAIT</b>	<p>Specifies how long (in seconds) the Apply program is to wait after it encounters an error before it retries the operation that failed.</p> <p><b>300</b> (default) The retry wait time is 300 seconds (five minutes).</p> <p><i>retry-wait-time</i> The wait time, entered as a number between 0 and 35000000 inclusive, before the Apply program retries the failed operation.</p>
<b>COPYONCE</b>	<p>Specifies whether the Apply program executes one copy cycle for each subscription set that is eligible at the time the Apply program is invoked. Then the Apply program terminates. An eligible subscription set meets the following criteria:</p> <ul style="list-style-type: none"> <li>• (ACTIVATE &gt; 0) in the subscription sets (IBMSNAP_SUBS_SET) table. When the ACTIVATE column value is greater than zero, the subscription set is active indefinitely or is used for a one-time-only subscription processing.</li> <li>• (REFRESH_TYPE = R or B) or (REFRESH_TYPE = E and the specified event occurred). The REFRESH_TYPE column value is stored in the IBMSNAP_SUBS_SET table.</li> </ul> <p>The MAX_SYNCH_MINUTES limit from the IBMSNAP_SUBS_SET table and the END_OF_PERIOD timestamp from the subscription events (IBMSNAP_SUBS_EVENT) table are honored if specified.</p> <p><b>*NO</b> (default) The Apply program does not execute one copy cycle for each eligible subscription set.</p> <p><b>*YES</b> The Apply program executes one copy cycle for each eligible subscription set and then terminates.</p>

Table 49. STRDPRAPY command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>TRLREUSE</b>	<p>Specifies whether the Apply program empties the Apply trail (IBMSNAP_APPLYTRAIL) table when the Apply program starts.</p> <p><b>*NO</b> (default) The Apply program does not empty the IBMSNAP_APPLYTRAIL table during program startup.</p> <p><b>*YES</b> The Apply program empties the IBMSNAP_APPLYTRAIL table during program startup.</p>
<b>OPTSNGSET</b>	<p>Specifies whether the performance of the Apply program is optimized if only one subscription set is processed. This parameter does not pertain to replica target tables.</p> <p>If you set this parameter to <b>*YES</b>, the Apply program fetches the members and columns of a subscription set only once and reuses this fetched information when processing the same subscription set in two or more consecutive processing cycles.</p> <p><b>*NO</b> (default) The performance of the Apply program is not optimized if only one subscription set is processed.</p> <p><b>*YES</b> The performance of the Apply program is optimized if only one subscription set is processed. The Apply program reuses the subscription set information during subsequent processing cycles, requiring less CPU resources and improving throughput rates.</p>

## Usage notes

You can set up the system to start the subsystem automatically by adding the command that is referred to in the QSTRUPPGM value on your system. If you use the QDP4/QZSNDPR subsystem, it is started as part of the **STRDPRAPY** command processing.

If the relational database (RDB) specified by the **CTLSVR** parameter is a DB2 Universal Database for iSeries database, the tables on the server are found in the ASN library. If the RDB is not a DB2 Universal Database for iSeries database, you can access the tables by using ASN as the qualifier.

### Error conditions when starting the Apply program

The **STRDPRAPY** command issues an error message if any of the following conditions occur:

- If the user does not exist.



- If the user running the command is not authorized to the user profile specified on the command or the job description.
- If an instance of the Apply program is already active on the local system for this combination of Apply qualifier and control server.
- If the RDB name specified by the **CTLSVR** parameter is not in the relational database directory.
- If the control tables do not exist on the RDB specified by the **CTLSVR** parameter.
- If no subscription sets are defined for the Apply qualifier specified by the **APYQUAL** parameter.

An Apply program must be started for each unique Apply qualifier in every subscription sets (IBMSNAP\_SUBS\_SET) table. You can start multiple Apply programs by specifying a different Apply qualifier each time that you issue the **STRDPRAPY** command. These Apply programs will run under the same user profile.

### Identifying Apply program jobs

Each Apply program is identified using both the Apply qualifier and the control server names. When run, the job started for the Apply program does not have sufficient external attributes to correctly identify which Apply program is associated with a particular Apply qualifier and control server combination. Therefore, the job is identified in the following way:

- The job is started under the user profile associated with the **USER** parameter.
- The first 10 characters of the Apply qualifier are truncated and become the job name.
- DB2 DataPropagator for iSeries maintains an Apply job (IBMSNAP\_APPLY\_JOB) table named in the ASN library on the local system. The table maps the Apply qualifier and control server values to the correct Apply program job.
- You can view the job log. The Apply qualifier and control server names are used in the call to the Apply program.

In general, you can identify the correct Apply program job by looking at the list of jobs running in the QZSNDPR subsystem if both:

- The first 10 characters of the Apply qualifier name are unique.
- The Apply program is started for the local control server only.

### Examples for STRDPRAPY

The following examples illustrate how to use the **STRDPRAPY** command.

## STRDPRAPY

### Example 1

To start the Apply program that uses the AQHR Apply qualifier and Apply control tables that reside locally and to generate a trace file that contains error and execution flow information:

```
STRDPRAPY APYQUAL(AQHR) CTLSVR(*LOCAL) TRACE(*ALL)
```

### Example 2

To start an Apply program with Apply control tables that reside locally and to specify that the job that started this Apply program automatically end when the Apply program has nothing left to process:

```
STRDPRAPY APYQUAL(AQHR) CTLSVR(*LOCAL) ALWINACT(*NO)
```

### Example 3

To start an Apply program that empties the IBMSNAP\_APPLYTRAIL table during program startup:

```
STRDPRAPY APYQUAL(AQHR) CTLSVR(*LOCAL) TRLREUSE(*YES)
```

### Example 4

To start an Apply program with all default values:

```
STRDPRAPY
```

### Related tasks:

- Chapter 10, “Operating the Apply program” on page 139

### Related reference:

- “asnapply: Starting Apply (UNIX, Windows, z/OS)” on page 308

---

## STRDPRCAP: Starting Capture (OS/400)

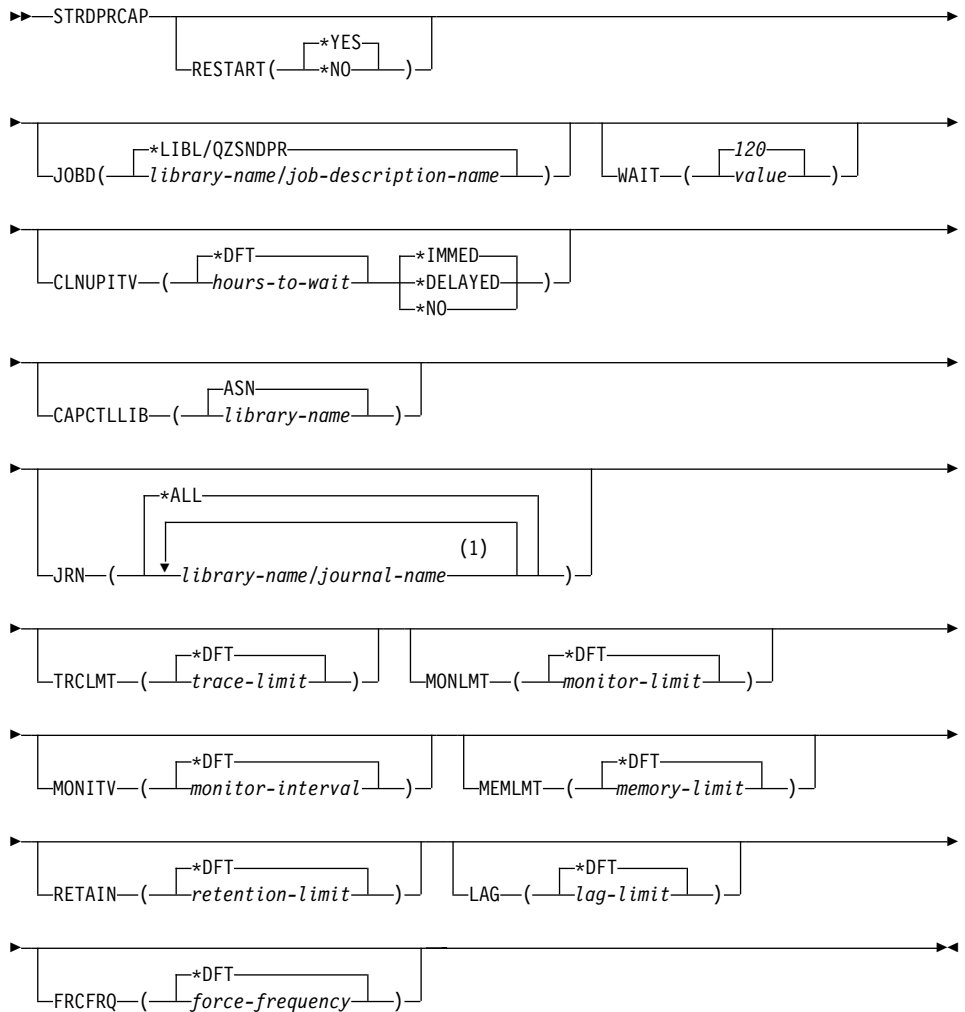
Use the Start DPR Capture (**STRDPRCAP**) command to start capturing changes to OS/400 database tables on iSeries servers. Because this command processes all replication sources in the register (IBMSNAP\_REGISTER) table, make sure that you are running this command with the proper authority.

After you start the Capture program, it runs continuously until you stop it or it detects an unrecoverable error.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

**To start the DPR Capture program using the STRDPRCAP command:**



### Notes:

- 1 You can specify up to 50 journals.

Table 50. STRDPRCAP command parameter definitions for OS/400

Parameter	Definition and prompts
RESTART	<p>Specifies how the Capture program handles warm and cold starts.</p> <p><b>*YES</b> (default) The Capture program continues processing the changes from the point where it was when it ended previously. This is also known as a <i>warm start</i> and is the normal mode of operation.</p> <p><b>*NO</b> The Capture program removes all information from the change-data (CD) tables. The Capture program also removes all information from the unit-of-work (UOW) table when you specify JRN(*ALL).  All subscriptions for affected source tables are fully refreshed before change capturing resumes. This process is also known as a <i>cold start</i>.  By specifying RESTART(*NO) and JRN(<i>library-name/journal-name</i>), you can cold start the Capture program for specified journals.</p>
JOBID	<p>Specifies the name of the job description to use when submitting the Capture program.</p> <p><b>*LIBL/QZSNDPR</b> (default) Specifies the default job description provided with DB2 DataPropagator for iSeries.</p> <p><i>library-name/job-description-name</i> The name of the job description used for the Capture program.</p>

Table 50. STRDPRCAP command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>WAIT</b>	<p>Specifies the maximum number of seconds (60 to 6 000) to wait before the Capture program checks its status. You can use this value to tune the responsiveness of the Capture program.</p> <p>A low value reduces the time that the Capture program takes before ending or initializing, but can have a negative effect on system performance. A higher value increases the time that the Capture program takes before ending or initializing, but can improve system performance. A value that is too high can result in decreased responsiveness while the Capture program is performing periodic processing. The amount of the decrease in responsiveness depends on the amount of change activity to source tables and the amount of other work occurring on the system.</p> <p><b>120</b> (default) The Capture program waits 120 seconds.</p> <p><i>value</i> The maximum number of seconds that the Capture program waits.</p>

Table 50. STRDPRCAP command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
CLNUPITV	<p>Specifies the maximum amount of time (in hours) before the Capture program prunes old records from the change-data (CD), unit-of-work (UOW), signal (IBMSNAP_SIGNAL), Capture monitor (IBMSNAP_CAPMON), Capture trace (IBMSNAP_CAPTRACE), and Apply qualifier cross-reference (IBMSNAP_AUTHTKN) tables.</p> <p>This parameter works with the <b>RETAIN</b> parameter to control pruning of the CD, UOW, IBMSNAP_SIGNAL, and IBMSNAP_AUTHTKN tables, with the <b>MONLMT</b> parameter to control pruning of the IBMSNAP_CAPMON table, and with the <b>TRCLMT</b> parameter to control pruning of the IBMSNAP_CAPTRACE table.</p> <p>(Use the <b>STRDPRCAP</b> command to set the <b>RETAIN</b>, <b>MONLMT</b>, and <b>TRCLMT</b> parameters for a Capture program. Use the <b>CHGDPRCAPA</b> or <b>OVRDPRCAPA</b> command to change these parameter settings.)</p> <p>There are two parts to the <b>CLNUPITV</b> parameter:</p> <p><b>*DFT</b> (default) The Capture program uses the value of the PRUNE_INTERVAL column from the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p><i>hours-to-wait</i> The pruning interval expressed as a specific number of hours (1 to 100).</p> <p><b>*IMMED</b> (default) The Capture program prunes old records at the beginning of the specified interval (or immediately), and at each interval thereafter.</p> <p><b>*DELAYED</b> The Capture program prunes old records at the end of the specified interval, and at each interval thereafter.</p> <p><b>*NO</b> The Capture program does not prune records.</p>

Table 50. STRDPRCAP command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>CAPCTLLIB</b>	<p>Specifies the Capture schema, which is the name of the library in which the Capture control tables reside.</p> <p><b>ASN</b> (default) The default library in which the Capture control tables reside.</p> <p><i>library-name</i> The name of the library in which the Capture control tables reside.</p>
<b>JRN</b>	<p>Specifies a subset of up to 50 journals that you want the Capture program to work with. The Capture program starts processing all the source tables that are currently journaled to this journal.</p> <p><b>*ALL</b> (default) The Capture program starts working with all of the journals that have any source tables journaled to them.</p> <p><i>library-name/journal-name</i> The qualified name of the journal that you want the Capture program to work with. When entering multiple journals, separate the journals with spaces.</p>
<b>TRCLMT</b>	<p>Specifies the trace limit (in minutes). The Capture program prunes any Capture trace (IBMSNAP_CAPTRACE) table rows that are older than the trace limit. The default is 10 080 minutes (seven days of trace entries).</p> <p><b>*DFT</b> (default) The Capture program uses the TRACE_LIMIT column value from the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p><i>trace-limit</i> The number of minutes of trace data kept in the IBMSNAP_CAPTRACE table after pruning.</p>

Table 50. STRDPRCAP command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>MONLMT</b>	<p>Specifies the monitor limit (in minutes). The Capture programs prunes any Capture monitor (IBMSNAP_CAPMON) table rows that are older than the monitor limit. The default is 10 080 minutes (seven days of monitor entries).</p> <p><b>*DFT (default)</b> The Capture program uses the MONITOR_LIMIT column value from the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p><i>monitor-limit</i> The number of minutes of monitor data kept in the IBMSNAP_CAPMON table after pruning.</p>
<b>MONITV</b>	<p>Specifies how frequently (in seconds) the Capture program inserts rows into the Capture monitor (IBMSNAP_CAPMON) table. The default is 300 seconds (five minutes).</p> <p><b>*DFT (default)</b> The Capture program uses the MONITOR_INTERVAL column value from the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p><i>monitor-interval</i> The number of seconds between row insertion into the IBMSNAP_CAPMON table. The monitor interval must be at least 120 seconds (two minutes). If you type a number that is less than 120, this parameter value is set to 120.</p>
<b>MEMLMT</b>	<p>Specifies the maximum size (in megabytes) of memory that the Capture journal job can use. The default is 32 megabytes.</p> <p><b>*DFT (default)</b> The Capture program uses the MEMORY_LIMIT column value from the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p><i>memory-limit</i> The maximum number of megabytes for memory.</p>



Table 50. STRDPRCAP command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>RETAIN</b>	<p>Specifies the new retention limit, which is the number of minutes that data is retained in the change-data (CD), unit-of-work (UOW), signal (IBMSNAP_SIGNAL), and Apply qualifier cross-reference (IBMSNAP_AUTHTKN) tables before it is removed. This value works with the <b>CLNUPITV</b> parameter value. When the <b>CLNUPITV</b> value is reached, the CD, UOW, IBMSNAP_SIGNAL, and IBMSNAP_AUTHTKN data is removed if this data is older than the retention limit.</p> <p>Ensure that the Apply intervals are set to copy changed information before the data reaches this <b>RETAIN</b> parameter value to prevent inconsistent data in your tables. If the data becomes inconsistent, the Apply program performs a full refresh.</p> <p>The default is 10 080 minutes (seven days). The maximum is 35000000 minutes.</p> <p><b>*DFT</b> (default) The Capture program uses the RETENTION_LIMIT column value from the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p><i>retention-limit</i> The number of minutes that the CD, UOW, IBMSNAP_SIGNAL, and IBMSNAP_AUTHTKN data is retained.</p>
<b>LAG</b>	<p>Specifies the new lag limit, which is the number of minutes that the Capture program can fall behind in processing before restarting.</p> <p>When the lag limit is reached (that is, when the timestamp of the journal entry is older than the current timestamp minus the lag limit), the Capture program initiates a cold start for the tables that it is processing in that journal. The Apply program then performs a full refresh to provide the Capture program with a new starting point.</p> <p>The default is 10 080 minutes (seven days). The maximum is 35000000 minutes.</p> <p><b>*DFT</b> (default) The Capture program uses the LAG_LIMIT column value from the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p><i>lag-limit</i> The number of minutes that the Capture program is allowed to fall behind.</p>

Table 50. STRDPRCAP command parameter definitions for OS/400 (continued)

Parameter	Definition and prompts
<b>FRCFRQ</b>	<p>Specifies how often (30 to 600 seconds) that the Capture program writes changes to the change-data (CD) and unit-of-work (UOW) tables. The Capture program makes these changes available to the Apply program either when the buffers are filled or when the FRCFRQ time limit expires, whichever is sooner.</p> <p>Use this parameter to make changes more readily available to the Apply program on servers with a low rate of source table changes. The <b>FRCFRQ</b> parameter value is a global value used for all defined source tables. Setting the <b>FRCFRQ</b> value to a low number can affect system performance.</p> <p>The default is 30 seconds.</p> <p><b>*DFT</b> (default) The Capture program uses the COMMIT_INTERVAL column value from the Capture parameters (IBMSNAP_CAPPARMS) table.</p> <p><i>force-frequency</i> The number of seconds that the Capture program keeps CD and UOW table changes in buffer space before making these changes available to the Apply program.</p>

Usage notes

The **CLNUPITV** parameter on the **STRDPRCAP** command specifies the maximum number of hours that the Capture program waits before pruning old records from the change-data (CD), unit-of-work (UOW), signal (IBMSNAP\_SIGNAL), Capture monitor (IBMSNAP\_CAPMON), Capture trace (IBMSNAP\_CAPTRACE), and Apply qualifier cross-reference (IBMSNAP\_AUTHTKN) tables.

You can run the **STRDPRCAP** command manually, or you can run the command automatically as a part of the initial program load (IPL startup program).

If the job description specified with the **JOB** parameter uses job queue QDP4/QZSNDR and the DB2 DataPropagator for iSeries subsystem is not active, the **STRDPRCAP** command starts the subsystem. If the job description is defined to use a different job queue and subsystem, you must start this subsystem manually with the Start Subsystem (**STRSBS**) command either before or after running the **STRDPRCAP** command:

```
STRSBS QDP4/QZSNDR
```

You can set up the system to start the subsystem automatically by adding the **STRSBS** command to the program that is referred to in the QSTRUPPGM system value on your system.

### Restarting Capture using warm or cold starts

The value of the **RESTART** parameter on the **STRDPRCAP** command controls how the Capture program handles warm and cold starts.

**Warm start process:** Warm start information is saved in most cases. Occasionally, warm start information is not saved. In this case, the Capture program uses the CD tables, UOW table, or the pruning control (IBMSNAP\_PRUNCNTL) table to resynchronize to the time that it was stopped.

**Automatic cold starts:** Sometimes the Capture program automatically switches to a cold start, even if you specified a warm start. On OS/400 systems, cold starts work on a journal-by-journal basis. So, for example, if a journal exceeds the lag limit, all replication sources using that journal are cold-started, whereas replication sources using a different journal are not cold started.

For more information about how the Capture program processes different journal entry types, see Table 100 on page 643.

## Examples for STRDPRCAP

The following examples illustrate how to use the **STRDPRCAP** command.

### Example 1

To initiate a warm start of a Capture program for two different journals:

```
STRDPRCAP RESTART(*YES) JRN(HR/QSQJRN ACCTS/QSQJRN)
```

### Example 2

To start a Capture program for one specified journal:

```
STRDPRCAP CAPCTLLIB(BSN) JRN(MARKETING/QSQJRN)
```

The Capture control tables reside under a library named BSN.

### Example 3

To start a Capture program without pruning for two journals:

```
STRDPRCAP RESTART(*YES) CLNUPITV(*DFT *NO) JRN(HR/QSQJRN ACCTS/QSQJRN)
```

### Example 4

To start a Capture program for one specified journal under the default Capture control library and to change the default trace limit pruning, monitor limit pruning, IBMSNAP\_CAPMON table insertion, and memory limit parameters:

## STRDPRCAP

STRDPRCAP CAPCTLLIB(ASN) JRN(SALES/QSQJRN) TRCLMT(1440) MONLMT(1440)  
MONITV(3600) MEMLMT(64)

### Example 5

To initiate a cold start of a Capture program:

STRDPRCAP RESTART(\*NO)

### Example 6

To start a Capture program with all default values:

STRDPRCAP

### Related tasks:

- Chapter 9, “Operating the Capture program” on page 117

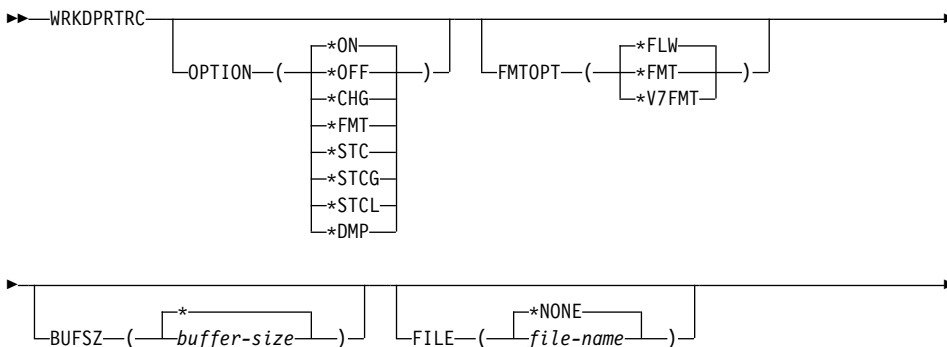
## WRKDPTRC: Using the DPR trace facility (OS/400)

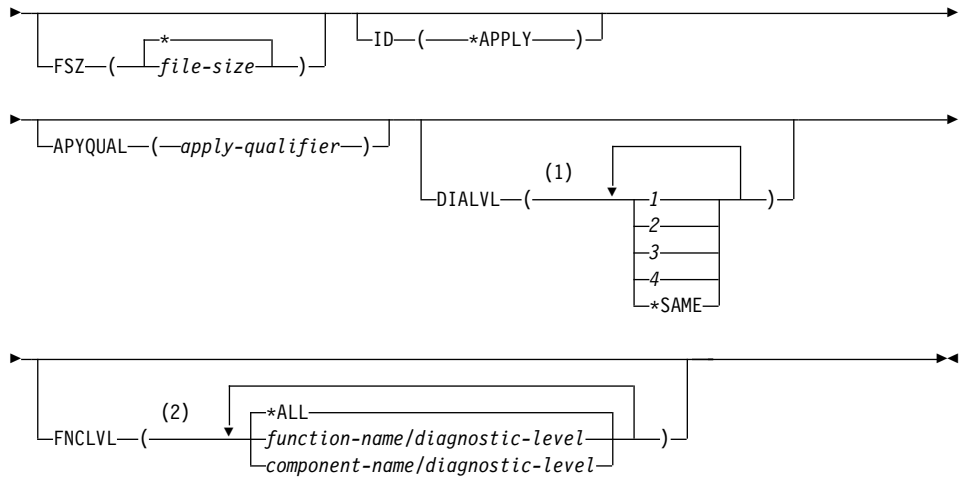
Use the DPR trace (**WRKDPRTRC**) command to run the trace facility. The trace facility logs program flow information for specified Apply programs. You can provide this trace information to IBM Software Support for troubleshooting assistance.

After you type the command name on the command line, you can press the F4 key to display the command syntax.

To display a complete description of this command and all of its parameters, move the cursor to the command at the top of the screen and press the F1 key. To display a description of a specific parameter, place the cursor on that parameter and press the F1 key.

***To run the DPR trace facility using the WRKDPTRC command:***



**Notes:**

- 1 You can specify multiple values.
- 2 You can specify up to 20 functions or components.

Table 51. WRKDPRTTC command parameter definitions for OS/400

Parameter	Definition
<b>OPTION</b>	Specify one trace facility function. <ul style="list-style-type: none"> <li><b>*ON</b> (default) Turn the trace facility on. This option automatically creates a shared memory segment for tracing.</li> <li><b>*OFF</b> Turn the trace facility off.</li> <li><b>*CHG</b> Change the values of the trace facility parameters.</li> <li><b>*FMT</b> Format the trace facility output from shared memory.</li> <li><b>*STC</b> Display the status of a trace facility. This status information includes the trace version, application version, number of entries, buffer size, amount of buffer used, status code, and program timestamp.  This parameter option is equivalent to the <b>stat</b> option of the <b>asnttc</b> command used on UNIX, Windows, and z/OS operating systems.</li> <li><b>*STCG</b> Display the status of a trace facility in Replication Center readable format.</li> <li><b>*STCL</b> Display the status of a trace facility with additional version level information. This additional information includes the service levels of each module in the application and appears as long strings of text.  This parameter option is equivalent to the <b>statlong</b> option of the <b>asnttc</b> command used on UNIX, Windows, and z/OS operating systems.</li> <li><b>*DMP</b> Write the current contents of the trace buffer to a file.</li> </ul> <p>When prompting on the <b>WRKDPRTTC</b> command, you can press the F4 key to see a list of trace options.</p>

Table 51. WRKDPRTTC command parameter definitions for OS/400 (continued)

Parameter	Definition
<b>FMTOPT</b>	<p>Specifies the options of the format ID and is used with the <b>OPTION(*FMT)</b> parameter.</p> <p><b>*FLW</b> (default) Display the flow of the function calls.</p> <p><b>*FMT</b> Display the format of the trace buffer or trace file. Shows all the detailed data.</p> <p><b>*V7FMT</b> Format the trace buffer or trace file information in Version 7 format.</p> <p>When prompting on the <b>WRKDPRTTC</b> command, you can press the F4 key to see a list of format options.</p>
<b>BUFSZ</b>	<p>Specifies the size (in bytes) of the trace buffer. You can enter an M, K, or G after the number to indicate megabytes, kilobytes, or gigabytes, respectively.</p> <p>The default is two megabytes.</p> <p><b>*</b> (default) Use the two megabyte default size.</p> <p><i>buffer-size</i> The buffer size in bytes.</p>
<b>FILE</b>	<p>Specifies whether the trace output is written to a file.</p> <p><b>*NONE</b> (default) The trace output goes to shared memory only.</p> <p><i>file-name</i> The name of the output file. If you are using the <b>OPTION(*DMP)</b> parameter, this file name represents the name of a dump file.</p>
<b>FSZ</b>	<p>Specifies the size (in bytes) of the file where the trace data is stored. You can enter an M, K, or G after the number to indicate megabytes, kilobytes, or gigabytes, respectively.</p> <p>The default is two gigabytes.</p> <p><b>*</b> (default) Use the two gigabyte default size.</p> <p><i>file-size</i> The file size in bytes.</p>

Table 51. WRKDPRTTC command parameter definitions for OS/400 (continued)

Parameter	Definition								
ID	<p>Specifies the type of program to be traced.</p> <p><b>*APPLY</b> (default) An Apply program trace.</p>								
APYQUAL	<p>Specifies the name of Apply program to be traced.</p> <p><i>apply-qualifier</i> The name of the Apply qualifier.</p>								
DIALVL	<p>Specifies the types of trace records to be recorded by the trace facility. Trace records are categorized by a diagnostic mask number:</p> <table><tr><td>1</td><td>Flow data, which includes the entry and exit points of functions.</td></tr><tr><td>2</td><td>Basic data, which includes all major events encountered by the trace facility.</td></tr><tr><td>3</td><td>Detailed data, which includes the major events with descriptions.</td></tr><tr><td>4</td><td>Performance data.</td></tr></table> <p><b>*SAME</b> This command uses the diagnostic level settings from the previous trace facility.</p> <p>You can enter one or more diagnostic mask numbers. The numbers that you enter must be in ascending order. Do not type spaces between the numbers.</p> <p><b>Important:</b> The number levels are <i>not</i> inclusive.</p> <p>When you start the trace facility, the default is DIALVL(1234). When you subsequently invoke the trace facility, the default is *SAME.</p> <p>When prompting on the <b>WRKDPRTTC</b> command, you can press the F4 key to see a list of available diagnostic levels.</p>	1	Flow data, which includes the entry and exit points of functions.	2	Basic data, which includes all major events encountered by the trace facility.	3	Detailed data, which includes the major events with descriptions.	4	Performance data.
1	Flow data, which includes the entry and exit points of functions.								
2	Basic data, which includes all major events encountered by the trace facility.								
3	Detailed data, which includes the major events with descriptions.								
4	Performance data.								



Table 51. WRKDPRTTC command parameter definitions for OS/400 (continued)

Parameter	Definition
FNCLVL	<p>Specifies if a particular function or component identifier is to be traced.</p> <p><b>*ALL</b> (default) All functions and components are included in the trace facility.</p> <p><i>function-name/diagnostic-level</i> The name of the function to be traced and the corresponding diagnostic mask numbers.</p> <p><i>component-name/diagnostic-level</i> The name of the component to be traced and the corresponding diagnostic mask numbers.</p> <p>You can enter up to 20 function or component names.</p>

## Examples for WRKDPRTTC

The following examples illustrate how to use the **WRKDPRTTC** command.

### Example 1

To start an Apply trace on the Apply qualifier AQ1 for all functions and components with output written to a file called TRCFILE:

```
WRKDPRTTC OPTION(*ON) FILE(TRCFILE) ID(*APPLY) APYQUAL(AQ1)
```

### Example 2

To end an Apply trace on the Apply qualifier AQ1:

```
WRKDPRTTC OPTION(*OFF) ID(*APPLY) APYQUAL(AQ1)
```

### Example 3

To change an Apply trace on the Apply qualifier AQ1 to diagnostic levels 3 and 4 (detailed and performance data) for all functions and components:

```
WRKDPRTTC OPTION(*CHG) ID(*APPLY) APYQUAL(AQ1) DIALVL(34)
```

### Example 4

To display the status of an Apply trace on the Apply qualifier AQ1:

```
WRKDPRTTC OPTION(*STC) ID(*APPLY) APYQUAL(AQ1)
```

### Example 5

To display the function calls on the Apply qualifier AQ1 at diagnostic levels 3 and 4:

```
WRKDPRTTC OPTION(*FMT) FMTOPT(*FLW) ID(*APPLY) APYQUAL(AQ1) DIALVL (34)
```

## WRKDPRTTC

### Example 6

To write the Apply trace information of the Apply qualifier AQ1 to a dump file named DMPFILE:

```
WRKDPRTTC OPTION(*DMP) FILE(DMPFILE) ID(*APPLY) APYQUAL(AQ1)
```

### Related reference:

- “asnttc: Operating the replication trace facility (UNIX, Windows, z/OS)” on page 341

---

## Chapter 19. Operating the replication programs (z/OS)

This chapter consists of the following sections:

- “Using JCL or system-started tasks to operate the replication programs (z/OS)”
- “Using MVS Automatic Restart Manager (ARM) to automatically restart replication programs (z/OS)” on page 455
- “Migrating your replication environment to data-sharing mode (z/OS)” on page 456

---

### Using JCL or system-started tasks to operate the replication programs (z/OS)

On z/OS, you can operate the Capture program, the Apply program, and the Replication Alert Monitor either with JCL or as system started tasks.

#### Using JCL to operate replication programs

This section describes how to use JCL to operate the Capture program, Apply program, and the Replication Alert Monitor.

You'll find sample JCL in the Program directory.

#### To start the Capture program on z/OS with JCL:

1. Customize the JCL in library SASNLJCL(ASNL2RN#), where # is the level of your DB2 for z/OS (for example, for Version 7, replace # with the numeral 7). Prepare the JCL for z/OS by specifying the appropriate optional invocation parameters in the PARM field of the Capture job. If you want to run in batch mode, you can also set time zone and language environment variables in the JCL.

The following example of this line in the invocation JCL includes setting the TZ and LANG variables:

```
//CAPJFA EXEC PGM=ASNCAP, PARM='ENVAR('TZ=PST8PDT','LANG=en_US')/  
DSN6 cold capture_schema=JFA autostop'
```

2. Submit the JCL from TSO or from the MVS console.

#### To start the Apply program on z/OS with JCL:

Prepare the JCL for z/OS by specifying the appropriate invocation parameters in the PARM field of the Apply job. Customize the JCL to meet your site's requirements. Invocation JCL in library SASNAJCL(ASNA2RN#), where # is the level of your DB2 for z/OS (for example, for Version 7, replace # with the numeral 7), is included with the Apply for z/OS product.

An example of this line in the invocation JCL is:

```
//apyasn EXEC PGM=ASNAPPLY,PARM='control_server=CTLDB1  
                                apply_qual=myqual spillfile=disk'
```

### To start the Replication Alert Monitor on z/OS with JCL:

Prepare the JCL for z/OS by specifying the appropriate invocation parameters in the PARM field of the Replication Alert Monitor job. Customize the JCL to meet your site's requirements. Invocation JCL in library SASNMJCL(ASNMON#) is included with the Replication Alert Monitor for z/OS product.

An example of this line in the invocation JCL is:

```
//monasn EXEC PGM=ASNMON,PARM='monitor_server=MONDB1  
                                monitor_qualifier=monqual'
```

### To modify started programs on z/OS with JCL:

After you start the Capture program, the Apply program, or the Replication Alert Monitor, you can use the MODIFY command to stop the program or to perform related tasks. You must run the MODIFY command from an MVS console. You can use the abbreviation F, as shown in the following syntax example:

(1)

► F—*jobname*—, — Parameters |—————►

#### Notes:

- 1 For descriptions of the parameters, see Chapter 17, “System commands for replication (UNIX, Windows, z/OS)” on page 303.

Basically, F *jobname* , replaces the actual command name: **asnacmd**, **asnccmd**, or **asnmcmd**. For example, to stop the Capture program you would use the following command:

```
F capjfa,stop
```

For information about MODIFY, see *z/OS MVS System Commands*.

## Using system-started tasks to operate replication programs

This section describes how to use system-started tasks to operate the Capture program, Apply program, and the Replication Alert Monitor.

### Setup to start the Capture for z/OS program as a system-started task:

1. Create a procedure (*procname*) in your PROCLIB. This procedure contains the JCL (for example ASNL2RN6, ASNL2RN7, or ASNL2RN8 in library SASNLJCL) required to run the Capture program.
2. Create an entry in the RACF STARTED class for the *procname*. This entry associates the *procname* with the RACF user ID to be used to start the Capture program. Make sure that the necessary DB2 authorization is granted to this user ID before you start the Capture program.
3. From the MVS system console, run the command **start *procname***.

---

## Using MVS Automatic Restart Manager (ARM) to automatically restart replication programs (z/OS)

The Capture program, the Apply program, and the Replication Alert Monitor can be used with the MVS Automatic Restart Manager (ARM). ARM is an MVS recovery function that can improve the availability of specific batch jobs or started tasks. When a job or task fails, or the system on which it is running fails, ARM can restart the job or task without operator intervention. ARM uses element names to identify the applications with which it works. Each MVS ARM enabled application generates a unique element name for itself that it uses in all communication with ARM. ARM tracks the element name and has its restart policy defined in terms of element names. For details about setting up ARM, see *z/OS MVS Sysplex Services Guide*, SA22-7617.

### Prerequisites:

Ensure that ARM is installed and that the replication programs are set up correctly. If you are going to use ARM with a replication program, ensure that the replication program is APF authorized. For example, to use ARM for the Apply program or the Replication Alert Monitor, you must copy the appropriate load module into an APF authorized library. (The Capture program must be APF authorized regardless of whether or not you are using ARM.)

When you configure ARM, use the following element names for the replication programs:

#### **Capture program**

ASNTCxxxxyyyy

#### **Apply program**

ASNTAxxxxyyyy

#### **Replication Alert Monitor**

ASNAMxxxxyyyy

where, xxxx is the DB2 subsystem name and yyyy is the data-sharing member name (the latter is needed only for data-sharing configurations). The element

name is always 16 characters long, padded with blanks. Element names must be unique in the entire sysplex; therefore, to use ARM, you may run only one instance of a particular program per subsystem.

The replication programs use the element name to register with ARM during initialization. They do not provide ARM with an event exit when they register. (The event exit is not needed because the replication programs do not run as an MVS subsystem.) ARM restarts registered programs for you if they terminate abnormally (for example, if a segment violation occurs). A registered replication program de-registers if it terminates normally (for example, due to a STOP command) or if it encounters an invalid registration.

**Tip:** If you start the Capture or Apply program using parameter NOTERM=Y, the program does not stop when DB2 is quiesced. In this case, the program does not de-register from ARM. It continues to run but does not capture data until DB2 is restarted.

---

## Migrating your replication environment to data-sharing mode (z/OS)

If the Capture program is running in non-data sharing mode but you migrate your installation to data-sharing mode, you must run the plexify utility (ASNPLXFY) once. Run this utility on the data sharing configuration before warm-starting the Capture program so that the Capture program starts at the correct LRSN. This utility migrates the data in the restart (IBMSNAP\_RESTART) table. It converts the non-data sharing log sequence numbers (RBA) to the equivalent sequence numbers (LRSN) in a data-sharing environment.

### Prerequisites:

Use either the same user ID that you use to run the Capture program, or one that has the same privileges. Ensure that the ASNPLXFY utility is APF authorized. The ASNPLXFY plan must be bound to the subsystem. Also, the subsystem must be running in data sharing mode. For details about binding this utility, see the Program Directory.

### Procedure:

To run the ASNPLXFY utility in the USS data-sharing environment:

1. Stop the Capture program.
2. Type the following command from a command line:

```
ASNPLXFY yoursystem captureschema
```

where the name of the subsystem is required and the Capture schema is optional. The default Capture schema is ASN.

3. Warm-start the Capture program.





---

## Chapter 20. Using the Windows Service Control Manager to issue system commands (Windows)

This section describes how to create services that start the replication programs on Windows operating systems. You can create services for each Capture control server, Apply control server, and Monitor control server. The services are grouped with other DB2 services. If you want to change the parameters for a program after the service is started, you must drop the service and create a new one.

- “Creating a replication service”
- “Operating a replication service” on page 461
- “Dropping a replication service” on page 461

---

### Creating a replication service

Before you create a replication service, make sure that the DB2 instance service is running. If the DB2 instance service is not running when you create the replication service, the replication service is created but it is not started automatically.

Use one of the following methods to create a replication service:

- In the Replication Center, specify the startup parameters for the program that you want to start and select **Start the *program\_name* program as a Windows service**. See the Replication Center help for details.
- Use the **asnsrct** command. See “asnsrct: Creating a DB2 Replication Service to start Capture, Apply, or the Replication Alert Monitor (Windows only)” on page 338 for command syntax and parameter descriptions.

**Tip:** If your replication service is set up correctly, the service name is sent to stdout after the service is started successfully. If the service does not start, check the log files for the program that you were trying to start. By default, the log files are in the directory specified by the DB2PATH environment variable. You can override this default by specifying the path parameter (**capture\_path**, **apply\_path**, **monitor\_path**) for the program that is started as a service. Also, you can use the Windows Service Control Manager (SCM) to view the status of the service.

When you create a service, you must specify the account name that you use to log on to Windows and the password for that account name.

You can add more than one replication service to your system. You can add one service for each schema on every Capture server, and for each qualifier on every Apply control server and Monitor control server, respectively. For example, if you have five databases and each database is an Apply control server, a Capture control server, and a Monitor control server, you can create fifteen replication services. If you have multiple schemas or qualifiers on each server, you could create more services.

When you create a replication service, it is added to the SCM in Automatic mode and the service is started. Windows registers the service under a unique service name and display name.

### Replication service name

The replication service name uniquely identifies each service and is what you use when you want to stop or start a service. It has the following format:

*DB2.instance.alias.program.qualifier\_or\_schema*

where:

- *instance* is the name of the DB2 instance.
- *alias* is the database alias of the Capture control server, Apply control server, or Monitor control server.
- *program* is one of the following values: CAP (for Capture program), APP (for Apply program), or MON (for Replication Alert Monitor program)
- *qualifier\_or\_schema* is one of the following identifiers: Apply qualifier, Monitor qualifier, or Capture schema

**Example:** The following service name is for a Capture program that has the schema ASN and is working with database DB1 under the instance called INST1:

DB2.INST1.DB1.CAP.ASN

### Display name for the replication service

The display name is a text string that you see in the Services window and it is a more readable form of the service name. For example:

DB2 - INST1 DB1 CAPTURE ASN

If you want to add a description for the service, use the Service Control Manager (SCM) after you create a replication service. You can also use the SCM to specify a user name and a password for a service.

---

## Operating a replication service

After you create a replication service, you can stop it and start it again.

Use one of the following methods to stop a service:

- SCM
- **net stop** command

**Important:** When you stop a replication service, the program associated with that service is stopped automatically. However, if you stop a program using a replication system command (**asnacmd**, **asnccmd**, or **asnmcmd**), the service that you used to start that program continues running until you stop it explicitly.

Use one of the following methods to start a service for replication commands:

- SCM
- **net start** command
- Replication Center

**Important:** If you started a replication program from a service, you will get an error if you try to start the program using the same schema or qualifier.

Depending on the Windows operating system that you're using, you can also modify the operation of the replication service:

- On Windows 2000, you can use the SCM to pause, resume, and restart the replication service.
- On Windows NT, you can use the SCM to pause or continue the replication service.

---

## Dropping a replication service

If you no longer need a replication service you can drop it so that it is removed from the SCM. Also, if you want to change the start-up parameters for a program that is started by a service, you must drop the service and then create a new one using new start-up parameters.

To drop a service for replication commands, use the **asnsdrop** command.

### Related reference:

- “asnsrct: Creating a DB2 Replication Service to start Capture, Apply, or the Replication Alert Monitor (Windows only)” on page 338
- “asnsdrop: Dropping a DB2 Replication Service (Windows only)” on page 340



---

## Chapter 21. Scheduling replication programs on various operating systems

You might want to schedule the Capture program, the Apply program, or the Replication Alert Monitor program to start at a prescribed time using the commands that are available on your operating system.

---

### Scheduling programs on UNIX operating systems

Use the **at** command to start a program at a specific time. For example, the following commands start the programs at 3:00 p.m. on Friday:

***Scheduling the Capture program:***

```
at 3pm Friday asncap autoprune=n
```

***Scheduling the Apply program:***

```
at 3pm Friday asnapply applyqual=myqual
```

***Scheduling the Replication Alert Monitor program:***

```
at 3pm Friday asnmon monitor_server=db2srv1 monitor_qualifier=mymon
```

---

### Scheduling programs on Windows operating systems

If you're not using the Windows Service Control Manager, you might want to use the **AT** command to start the programs at a specific time. Before you enter the **AT** command, start the Windows Schedule Service.

The following examples start each program at 15:00 (3:00 p.m.):

***Scheduling the Capture program:***

```
c:\>AT 15:00 /interactive "c:\SQLLIB\BIN\db2cmd.exe c:\CAPTURE\asncap.exe"
```

***Scheduling the Apply program:***

```
c:\>AT 15:00 /interactive "c:\SQLLIB\BIN\db2cmd.exe  
c:\SQLLIB\BIN\asnapply.exe control_server=cntldb apply_qual=qualid1"
```

***Scheduling the Replication Alert Monitor program:***

```
c:\>AT 15:00 /interactive "c:\SQLLIB\BIN\db2cmd.exe  
c:\CAPTURE\asnmon.exe monitor_server=db2srv1 monitor_qualifier=mymon"
```

---

## Scheduling programs on z/OS operating systems

Use either the **\$TA JES2** command or the **AT NetView** command to start Capture for z/OS at a specific time.

**To schedule a program on z/OS:**

1. Create a procedure that calls the program for z/OS in the PROCLIB.
2. Modify the ICHRIN03 RACF module (or appropriate definitions for your MVS security package) to associate the procedure with a user ID.
3. Link-edit the module in SYS1.LPALIB.

See *MVS/ESA JES2 Commands* for more information about using the **\$TA JES2** command, and the *NetView for MVS Command Reference* for more information about using the **AT NetView** command.

---

## Scheduling programs on the OS/400 operating system

Use the **ADDJOBSCDE** command to start the Apply program at a specific time.

Use the **SBMJOB** command to schedule the start of the Capture program on OS/400:

```
SBMJOB CMD('STRDPRCAP...')SCDDATE(...)SCDTIME(...)
```

---

## Chapter 22. How the DB2 replication components communicate

The replication components run independently of each other, and they rely on information that they each store in the replication control tables to communicate with each other. DB2® replication has four components:

- The Replication Center
- The Capture program or triggers
- The Apply program
- The Replication Alert Monitor

The Replication Center stores the initial information about registered sources, subscription sets, and alert conditions in the control tables. The Capture program, the Apply program, and the Capture triggers update the control tables to indicate the progress of replication and to coordinate the processing of changes. The Replication Alert Monitor reads the control tables that have been updated by the Capture program, Apply program, and the Capture triggers to understand the problems and progress at a server.

---

### The Replication Center, the Capture program or triggers, and the Apply program

When you register a table, view, or nickname as a replication source, the Replication Center creates an SQL script that stores the information for this source in the replication control table that contains all registration information, the register (IBMSNAP\_REGISTER) table. The SQL script generated by the Replication Center also creates the CD tables for the registered sources.

The IBMSNAP\_REGISTER table contains one row for every registered source table, and one row for every underlying table in a registered view. This table contains the following kinds of information about each registered source:

- The schema name and name of the source table
- The structure type of each registered source table
- The schema name and name of the CD table
- For registered views, the names of the CD tables for the underlying tables in this view (if the underlying tables are registered)
- The schema name and name of the internal CCD table, if there is one
- The conflict-detection level for update-anywhere sources

The Capture and Apply programs use the information in the IBMSNAP\_REGISTER table to communicate their respective status to each

other. This table has several more columns for related information. See the “*schema*.IBMSNAP\_REGISTER” on page 498 for more information about this table.

For OS/400® sources, including tables that are journaled remotely, there is also an extension to the IBMSNAP\_REGISTER table, IBMSNAP\_REG\_EXT, which contains extra information that is unique to iSeries™ systems, for example, the journal library and the journal name.

When you create a subscription set and add members to it, the Replication Center creates an SQL script that stores the information for this subscription set in the replication control tables that contain all subscription-set information: the subscription set (IBMSNAP\_SUBS\_SET), subscription-set member (IBMSNAP\_SUBS\_MEMBR), subscription-set columns (IBMSNAP\_SUBS\_COLS), and subscription-set statements (IBMSNAP\_SUBS\_STMTS) tables. The SQL script generated by the Replication Center also creates the target tables if they do not already exist.

The main subscription-set table, IBMSNAP\_SUBS\_SET, contains one row for every subscription set. This table contains the following kinds of information about each subscription set:

- The Apply qualifier
- The name of the subscription set
- The type of subscription set: read only or read/write (update anywhere)
- The names and aliases of the source and target databases
- The timing for processing the subscription set
- The current status for the subscription set

This table has several more columns for related information. See the “ASN.IBMSNAP\_SUBS\_SET” on page 526 for more information about this table.

The other subscription-set tables contain information about the subscription-set members, columns, and SQL statements (or stored procedures) that are processed with the set.

---

## The Capture program and the Apply program

The Capture program uses some of the replication control tables to indicate what changes have been made to the source database, and the Apply program uses these control-table values to detect what needs to be copied to the target database. The Capture program does not capture any information until the Apply program signals it to do so, and the Apply program does not signal the Capture program to start capturing changes until you define a replication source and associated subscription sets.



The following process describes how the Apply and Capture programs communicate, in a *typical* replication scenario, to ensure data integrity:

***Capturing data from a source database***

1. The Capture program reads the IBMSNAP\_REGISTER table during startup to determine which registered replication sources it must capture changes for, and it holds the registration information in memory.
2. The Capture program reads the DB2 log or journal continuously to detect change records (INSERT, UPDATE, and DELETE) for registered source tables or views. It also detects inserts to the signal (IBMSNAP\_SIGNAL) table in order to pick up signal actions that have been initialized by the Apply program or a user. When the Apply program inserts a CAPSTART signal in the IBMSNAP\_SIGNAL table and the Capture program detects the committed signal, the Capture program initializes the registration and starts capturing changes for the associated source.
3. Once the Capture program has started capturing changes for a registered source, the program writes one row (or two rows if you specified that updates should be saved as DELETE and INSERT statements) to the CD table for each *committed* change that it finds in the DB2 log or journal. The Capture program keeps uncommitted changes in memory until the changes are committed or aborted. Each registered replication source that is not an external CCD table has an associated CD table.
4. At each commit interval, the Capture program commits the data that it has written to the CD and UOW tables, and also updates the IBMSNAP\_REGISTER table to flag which CD tables have new committed changes.

***Applying data to a target database***

5. For all newly defined subscription sets, the Apply program first signals the Capture program to start capturing changes. Then a full refresh is performed for each member of the set (unless it is not a complete target table).
6. When any subscription set is eligible for replication, the Apply program checks the IBMSNAP\_REGISTER table to determine whether there are changes that need to be replicated.
7. The Apply program copies the changes from the CD table to the target table.
8. The Apply program updates the IBMSNAP\_SUBS\_SET table to record how much data the Apply program copied for each subscription set.
9. The Apply program updates the prune set (IBMSNAP\_PRUNE\_SET) table with a value that indicates the point to which it has read changes from the CD table.

***Pruning the CD tables***

10. When the Capture program prunes the CD tables, it uses the information in the IBMSNAP\_PRUNE\_SET table to determine which changes were applied, and deletes these already-replicated changes from the CD table.

---

## The Capture triggers and the Apply program

The Capture triggers use some of the replication control tables to indicate what changes have been made to the source database, and the Apply program uses these control-table values to detect what needs to be copied to the target database.

The Capture triggers start capturing information immediately. Unlike the Capture program, they do not wait for a signal from the Apply program.

The following process describes how the Capture triggers and the Apply program communicate, in a *typical* replication scenario, to ensure data integrity:

### ***Capturing data from a source***

1. Whenever a DELETE, UPDATE, or INSERT operation occurs at the registered replication source table, a Capture trigger records the change in the CCD table for that source table.

### ***Applying data to a target***

2. For all newly defined subscription sets, the Apply program first signals the Capture triggers to mark a valid starting point within the CCD table from which to start fetching changed data. Then a full refresh is performed for each member of the set (unless it is not a complete target table).
3. When the Apply program processes a subscription set for a non-DB2 relational source, it updates the register synchronization (IBMSNAP\_REG\_SYNCH) table, which causes an UPDATE trigger on that table to fire. The trigger updates the SYNCHPOINT value in the IBMSNAP\_REGISTER table to mark the highest SYNCHPOINT value in the CCD tables that it copied to the targets. During the next cycle, the Apply program will process new data in the CCD table that has a SYNCHPOINT value that is less than or equal to this SYNCHPOINT. Because the IBMSNAP\_REG\_SYNCH table is in the non-DB2 database, the Apply program writes to the table using the nickname for it that was created by the Replication Center.
4. The Apply program checks the IBMSNAP\_REGISTER table to determine whether there are changes that need to be replicated.
5. The Apply program copies the changes from the CCD table to the target table.
6. The Apply program updates the subscription set (IBMSNAP\_SUBS\_SET) table to record how much data the Apply program copied for each subscription set.

7. The Apply program updates the prune control (IBMSNAP\_PRUNCNTL) table for each registered source with a value that indicates the point to which it has read changes from the CCD table.

***Pruning the CCD tables***

8. The UPDATE trigger on the IBMSNAP\_PRUNCNTL table checks all of the CCD tables in the source database, and deletes the already-replicated changes from the CCD table.

---

## **The Replication Center and the Replication Alert Monitor**

When you define an alert condition with contacts who will be notified when the condition occurs, the Replication Center creates an SQL script that stores the information for this alert condition and its contacts in the replication control tables that contain all alert-condition and notification information: the Monitor conditions (IBMSNAP\_CONDITIONS), Monitor contacts (IBMSNAP\_CONTACTS), Monitor groups (IBMSNAP\_GROUPS), and Monitor group contacts (IBMSNAP\_CONTACTGRP) tables.

The main monitor alert table, the Monitor conditions table, contains one row for each condition that you want to be monitored. The table contains the following kinds of information about each alert condition:

- The Monitor qualifier
- The name and aliases of the Capture server or Apply server you want monitored
- The component that you want monitored (the Capture program or the Apply program)
- The Capture schema or Apply qualifier
- The name of the subscription set (if you want to monitor a set)
- The alert condition that you want monitored
- The contact to be notified if the condition occurs

This table has several more columns for related information. See “ASN.IBMSNAP\_CONDITIONS” on page 534 for more information about this table.

The other tables for the Replication Alert Monitor contain information about who will be notified if the alert condition occurs (either an individual contact or a group of contacts), how that contact will be notified (through e-mail or pager), and how often the contact will be notified if the condition continues.

---

## The Replication Alert Monitor, the Capture program, and the Apply program

The Replication Alert Monitor uses some of the Capture control tables to monitor the Capture program, and uses some of the Apply control tables to monitor the Apply program. It reads from different replication control tables at each Capture control server or Apply control server, depending on what it is monitoring. The Replication Alert Monitor does not interfere or communicate with the Capture or Apply program.

The following process describes how the Replication Alert Monitor monitors conditions for the Capture or Apply program and notifies contacts when the alert condition occurs:

1. The Replication Alert Monitor reads the alert conditions and the contact for each condition (for a Monitor qualifier) in the Monitor conditions (IBMSNAP\_CONDITIONS) table.
2. For each Capture control server or Apply control server that has a defined alert condition, the Replication Alert Monitor performs the following tasks:
  - a. The Replication Alert Monitor connects to the server and reads the replication control tables associated with each alert condition for that server to see if any of the conditions are met.
  - b. If any condition is met, the Replication Alert Monitor stores the data that is related to that condition in memory and continues processing the remaining alert conditions for that server.
  - c. When it is finished processing all the alert conditions for that server, the Replication Alert Monitor disconnects from the Capture control or Apply control server, inserts the alerts in the Monitor alerts (IBMSNAP\_ALERTS) table, and notifies the contacts for that condition.

### **Related concepts:**

- Chapter 14, “Using the DB2 Replication Center” on page 243

### **Related reference:**

- “List of tables used at the Apply control server” on page 479
- “List of tables used at the Capture control server” on page 476
- “List of tables used at the Monitor control server” on page 481

---

## Chapter 23. Table structures

This chapter describes the relational database tables that are used for replication at each server: the Capture control server, Apply control server, Monitor control server, and target server. The chapter provides three ways for you to reference the tables:

- The “Tables at a glance” on page 472 section provides quick reference sheets, which include the list of tables for the Capture control server, Apply control server, and Monitor control server, the columns in each table, and the indexes on each table.
- For an overview of the tables at each server, see:
  - “List of tables used at the Capture control server” on page 476
  - “List of tables used at the Apply control server” on page 479
  - “List of tables used at the Monitor control server” on page 481
  - “List of tables used at the target server” on page 482
- For a more detailed description about the tables at each server and a description of the columns in each table, see:
  - “Tables at the Capture control server and their column descriptions” on page 483
  - “Tables at the Apply control server and their column descriptions” on page 513
  - “Tables at the Monitor control server and their column descriptions” on page 533
  - “Tables at the target server and their column descriptions” on page 541.

In each section, the control tables are listed in alphabetical order by the actual table name (for example, ASN.IBMSNAP\_APPLYTRACE), and the target tables are listed in alphabetical order by their English table name (for example, replica table). The columns for each table are listed in the order in which they appear in the table.

Some of the control tables require that you *not* use SQL to update them (see particular table descriptions for details). Altering control tables inappropriately can cause problems such as unexpected results, loss of data, and reduced replication performance.

## Tables at a glance

Figure 15 and Figure 16 on page 473 show the tables at the Capture control server, the columns in each table, and the indexes on each table. Figure 18 on page 475 and Figure 17 on page 474 show the tables at the Apply control server, the columns in each table, and the indexes on each table. Figure 19 on page 476 shows the tables at the Monitor control server, the columns in each table, and the indexes on each table.

### Control tables used at the Capture control server (image 1 of 2)

<p><i>OS/400 only</i></p> <p><b>schema.IBMSNAP_AUTHTKN</b> (no unique index)</p> <table> <tr> <td>APPLY_QUAL</td><td>CHAR(18) NOT NULL</td></tr> <tr> <td>IBMSNAP_AUTHTKN</td><td>CHAR(26) NOT NULL</td></tr> <tr> <td>JRN_LIB</td><td>CHAR(10) NOT NULL</td></tr> <tr> <td>JRN_NAME</td><td>CHAR(10) NOT NULL</td></tr> <tr> <td>IBMSNAP_LOGMARKER</td><td>TIMESTAMP NOT NULL</td></tr> </table>	APPLY_QUAL	CHAR(18) NOT NULL	IBMSNAP_AUTHTKN	CHAR(26) NOT NULL	JRN_LIB	CHAR(10) NOT NULL	JRN_NAME	CHAR(10) NOT NULL	IBMSNAP_LOGMARKER	TIMESTAMP NOT NULL	<p><b>ASN.IBMSNAP_CAPSCHEMAS</b> (CAP_SCHEMA_NAME)</p> <table> <tr> <td>CAP_SCHEMA_NAME</td><td>VARCHAR(30)</td></tr> </table> <p><i>OS/400</i></p> <p><b>ASN.IBMSNAP_CAPSCHEMAS</b> (CAP_SCHEMA_NAME)</p> <table> <tr> <td>CAP_SCHEMA_NAME</td><td>VARCHAR(30)</td></tr> <tr> <td>STATUS</td><td>CHAR(1)</td></tr> </table>	CAP_SCHEMA_NAME	VARCHAR(30)	CAP_SCHEMA_NAME	VARCHAR(30)	STATUS	CHAR(1)																																		
APPLY_QUAL	CHAR(18) NOT NULL																																																		
IBMSNAP_AUTHTKN	CHAR(26) NOT NULL																																																		
JRN_LIB	CHAR(10) NOT NULL																																																		
JRN_NAME	CHAR(10) NOT NULL																																																		
IBMSNAP_LOGMARKER	TIMESTAMP NOT NULL																																																		
CAP_SCHEMA_NAME	VARCHAR(30)																																																		
CAP_SCHEMA_NAME	VARCHAR(30)																																																		
STATUS	CHAR(1)																																																		
<p><i>UNIX, Windows, and z/OS only</i></p> <p><b>schema.IBMSNAP_CAPENQ</b> (no unique index)</p> <table> <tr> <td>LOCKNAME</td><td>CHAR(9)</td></tr> </table>	LOCKNAME	CHAR(9)																																																	
LOCKNAME	CHAR(9)																																																		
<p><b>schema.IBMSNAP_CAPMON</b> (MONITOR_TIME)</p> <table> <tr> <td>MONITOR_TIME</td><td>TIMESTAMP NOT NULL</td></tr> <tr> <td>RESTART_TIME</td><td>TIMESTAMP NOT NULL</td></tr> <tr> <td>CURRENT_MEMORY</td><td>INT NOT NULL</td></tr> <tr> <td>CD_ROWS_INSERTED</td><td>INT NOT NULL</td></tr> <tr> <td>RECAP_ROWS_SKIPPED</td><td>INT NOT NULL</td></tr> <tr> <td>TRIGR_ROWS_SKIPPED</td><td>INT NOT NULL</td></tr> <tr> <td>CHG_ROWS_SKIPPED</td><td>INT NOT NULL</td></tr> <tr> <td>TRANS_PROCESSED</td><td>INT NOT NULL</td></tr> <tr> <td>TRANS_SPILLED</td><td>INT NOT NULL</td></tr> <tr> <td>MAX_TRAN_SIZE</td><td>INT NOT NULL</td></tr> <tr> <td>LOCKING_RETRIES</td><td>INT NOT NULL</td></tr> <tr> <td>JRN_LIB</td><td>CHAR(10)</td></tr> <tr> <td>JRN_NAME</td><td>CHAR(10)</td></tr> <tr> <td>LOGREADLIMIT</td><td>INT NOT NULL</td></tr> <tr> <td>CAPTURE_IDLE</td><td>INT NOT NULL</td></tr> <tr> <td>SYNCHTIME</td><td>TIMESTAMP NOT NULL</td></tr> </table>	MONITOR_TIME	TIMESTAMP NOT NULL	RESTART_TIME	TIMESTAMP NOT NULL	CURRENT_MEMORY	INT NOT NULL	CD_ROWS_INSERTED	INT NOT NULL	RECAP_ROWS_SKIPPED	INT NOT NULL	TRIGR_ROWS_SKIPPED	INT NOT NULL	CHG_ROWS_SKIPPED	INT NOT NULL	TRANS_PROCESSED	INT NOT NULL	TRANS_SPILLED	INT NOT NULL	MAX_TRAN_SIZE	INT NOT NULL	LOCKING_RETRIES	INT NOT NULL	JRN_LIB	CHAR(10)	JRN_NAME	CHAR(10)	LOGREADLIMIT	INT NOT NULL	CAPTURE_IDLE	INT NOT NULL	SYNCHTIME	TIMESTAMP NOT NULL																			
MONITOR_TIME	TIMESTAMP NOT NULL																																																		
RESTART_TIME	TIMESTAMP NOT NULL																																																		
CURRENT_MEMORY	INT NOT NULL																																																		
CD_ROWS_INSERTED	INT NOT NULL																																																		
RECAP_ROWS_SKIPPED	INT NOT NULL																																																		
TRIGR_ROWS_SKIPPED	INT NOT NULL																																																		
CHG_ROWS_SKIPPED	INT NOT NULL																																																		
TRANS_PROCESSED	INT NOT NULL																																																		
TRANS_SPILLED	INT NOT NULL																																																		
MAX_TRAN_SIZE	INT NOT NULL																																																		
LOCKING_RETRIES	INT NOT NULL																																																		
JRN_LIB	CHAR(10)																																																		
JRN_NAME	CHAR(10)																																																		
LOGREADLIMIT	INT NOT NULL																																																		
CAPTURE_IDLE	INT NOT NULL																																																		
SYNCHTIME	TIMESTAMP NOT NULL																																																		
<p><b>schema.IBMSNAP_CAPPARMS</b> (no unique index)</p> <table> <tr> <td>RETENTION_LIMIT</td><td>INT</td></tr> <tr> <td>LAG_LIMIT</td><td>INT</td></tr> <tr> <td>COMMIT_INTERVAL</td><td>INT</td></tr> <tr> <td>PRUNE_INTERVAL</td><td>INT</td></tr> <tr> <td>TRACE_LIMIT</td><td>INT</td></tr> <tr> <td>MONITOR_LIMIT</td><td>INT</td></tr> <tr> <td>MONITOR_INTERVAL</td><td>INT</td></tr> <tr> <td>MEMORY_LIMIT</td><td>SMALLINT</td></tr> <tr> <td>REMOTE_SRC_SERVER</td><td>CHAR(18)</td></tr> <tr> <td>AUTOPRUNE</td><td>CHAR(1)</td></tr> <tr> <td>TERM</td><td>CHAR(1)</td></tr> <tr> <td>AUTOSTOP</td><td>CHAR(1)</td></tr> <tr> <td>LOGREUSE</td><td>CHAR(1)</td></tr> <tr> <td>LOGSDOUT</td><td>CHAR(1)</td></tr> <tr> <td>SLEEPINTERVAL</td><td>SMALLINT</td></tr> <tr> <td>CAPTURE_PATH</td><td>VARCHAR(1040)</td></tr> <tr> <td>STARTMODE</td><td>VARCHAR(10)</td></tr> </table>	RETENTION_LIMIT	INT	LAG_LIMIT	INT	COMMIT_INTERVAL	INT	PRUNE_INTERVAL	INT	TRACE_LIMIT	INT	MONITOR_LIMIT	INT	MONITOR_INTERVAL	INT	MEMORY_LIMIT	SMALLINT	REMOTE_SRC_SERVER	CHAR(18)	AUTOPRUNE	CHAR(1)	TERM	CHAR(1)	AUTOSTOP	CHAR(1)	LOGREUSE	CHAR(1)	LOGSDOUT	CHAR(1)	SLEEPINTERVAL	SMALLINT	CAPTURE_PATH	VARCHAR(1040)	STARTMODE	VARCHAR(10)	<p><b>schema.IBMSNAP_CAPTRACE</b> (TRACE_TIME)</p> <table> <tr> <td>OPERATION</td><td>CHAR(8) NOT NULL</td></tr> <tr> <td>TRACE_TIME</td><td>TIMESTAMP NOT NULL</td></tr> <tr> <td>DESCRIPTION</td><td>VARCHAR(1024) NOT NULL</td></tr> </table> <p><i>OS/400</i></p> <p><b>schema.IBMSNAP_CAPTRACE</b> (TRACE_TIME)</p> <table> <tr> <td>OPERATION</td><td>CHAR(8) NOT NULL</td></tr> <tr> <td>TRACE_TIME</td><td>TIMESTAMP NOT NULL</td></tr> <tr> <td>JOB_NAME</td><td>CHAR(26) NOT NULL</td></tr> <tr> <td>JOB_STR_TIME</td><td>TIMESTAMP NOT NULL</td></tr> <tr> <td>DESCRIPTION</td><td>VARCHAR(298) NOT NULL</td></tr> </table>	OPERATION	CHAR(8) NOT NULL	TRACE_TIME	TIMESTAMP NOT NULL	DESCRIPTION	VARCHAR(1024) NOT NULL	OPERATION	CHAR(8) NOT NULL	TRACE_TIME	TIMESTAMP NOT NULL	JOB_NAME	CHAR(26) NOT NULL	JOB_STR_TIME	TIMESTAMP NOT NULL	DESCRIPTION	VARCHAR(298) NOT NULL
RETENTION_LIMIT	INT																																																		
LAG_LIMIT	INT																																																		
COMMIT_INTERVAL	INT																																																		
PRUNE_INTERVAL	INT																																																		
TRACE_LIMIT	INT																																																		
MONITOR_LIMIT	INT																																																		
MONITOR_INTERVAL	INT																																																		
MEMORY_LIMIT	SMALLINT																																																		
REMOTE_SRC_SERVER	CHAR(18)																																																		
AUTOPRUNE	CHAR(1)																																																		
TERM	CHAR(1)																																																		
AUTOSTOP	CHAR(1)																																																		
LOGREUSE	CHAR(1)																																																		
LOGSDOUT	CHAR(1)																																																		
SLEEPINTERVAL	SMALLINT																																																		
CAPTURE_PATH	VARCHAR(1040)																																																		
STARTMODE	VARCHAR(10)																																																		
OPERATION	CHAR(8) NOT NULL																																																		
TRACE_TIME	TIMESTAMP NOT NULL																																																		
DESCRIPTION	VARCHAR(1024) NOT NULL																																																		
OPERATION	CHAR(8) NOT NULL																																																		
TRACE_TIME	TIMESTAMP NOT NULL																																																		
JOB_NAME	CHAR(26) NOT NULL																																																		
JOB_STR_TIME	TIMESTAMP NOT NULL																																																		
DESCRIPTION	VARCHAR(298) NOT NULL																																																		
	<p><b>schema.IBMSNAP_PRUNCNTL</b> (SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL, APPLY_QUAL, SET_NAME, TARGET_SERVER, TARGET_TABLE, TARGET_OWNER, MAP_ID)</p> <table> <tr> <td>TARGET_SERVER</td><td>CHAR(18) NOT NULL</td></tr> <tr> <td>TARGET_OWNER</td><td>VARCHAR(30) NOT NULL</td></tr> <tr> <td>TARGET_TABLE</td><td>VARCHAR(128) NOT NULL</td></tr> <tr> <td>SYNCHTIME</td><td>TIMESTAMP</td></tr> <tr> <td>SYNCHPOINT</td><td>CHAR(10) FOR BIT DATA</td></tr> <tr> <td>SOURCE_OWNER</td><td>VARCHAR(30) NOT NULL</td></tr> <tr> <td>SOURCE_TABLE</td><td>VARCHAR(128) NOT NULL</td></tr> <tr> <td>SOURCE_VIEW_QUAL</td><td>SMALLINT NOT NULL</td></tr> <tr> <td>APPLY_QUAL</td><td>CHAR(18) NOT NULL</td></tr> <tr> <td>SET_NAME</td><td>CHAR(18) NOT NULL</td></tr> <tr> <td>CNTL_SERVER</td><td>CHAR(18) NOT NULL</td></tr> <tr> <td>TARGET_STRUCTURE</td><td>SMALLINT NOT NULL</td></tr> <tr> <td>CNTL_ALIAS</td><td>CHAR(8)</td></tr> <tr> <td>PHYS_CHANGE_OWNER</td><td>VARCHAR(30)</td></tr> <tr> <td>PHYS_CHANGE_TABLE</td><td>VARCHAR(128)</td></tr> <tr> <td>MAP_ID</td><td>VARCHAR(10) NOT NULL</td></tr> </table>	TARGET_SERVER	CHAR(18) NOT NULL	TARGET_OWNER	VARCHAR(30) NOT NULL	TARGET_TABLE	VARCHAR(128) NOT NULL	SYNCHTIME	TIMESTAMP	SYNCHPOINT	CHAR(10) FOR BIT DATA	SOURCE_OWNER	VARCHAR(30) NOT NULL	SOURCE_TABLE	VARCHAR(128) NOT NULL	SOURCE_VIEW_QUAL	SMALLINT NOT NULL	APPLY_QUAL	CHAR(18) NOT NULL	SET_NAME	CHAR(18) NOT NULL	CNTL_SERVER	CHAR(18) NOT NULL	TARGET_STRUCTURE	SMALLINT NOT NULL	CNTL_ALIAS	CHAR(8)	PHYS_CHANGE_OWNER	VARCHAR(30)	PHYS_CHANGE_TABLE	VARCHAR(128)	MAP_ID	VARCHAR(10) NOT NULL																		
TARGET_SERVER	CHAR(18) NOT NULL																																																		
TARGET_OWNER	VARCHAR(30) NOT NULL																																																		
TARGET_TABLE	VARCHAR(128) NOT NULL																																																		
SYNCHTIME	TIMESTAMP																																																		
SYNCHPOINT	CHAR(10) FOR BIT DATA																																																		
SOURCE_OWNER	VARCHAR(30) NOT NULL																																																		
SOURCE_TABLE	VARCHAR(128) NOT NULL																																																		
SOURCE_VIEW_QUAL	SMALLINT NOT NULL																																																		
APPLY_QUAL	CHAR(18) NOT NULL																																																		
SET_NAME	CHAR(18) NOT NULL																																																		
CNTL_SERVER	CHAR(18) NOT NULL																																																		
TARGET_STRUCTURE	SMALLINT NOT NULL																																																		
CNTL_ALIAS	CHAR(8)																																																		
PHYS_CHANGE_OWNER	VARCHAR(30)																																																		
PHYS_CHANGE_TABLE	VARCHAR(128)																																																		
MAP_ID	VARCHAR(10) NOT NULL																																																		
	<p><b>schema.IBMSNAP_PRUNE_LOCK</b> (no unique index)</p> <table> <tr> <td>DUMMY</td><td>CHAR(1)</td></tr> </table>	DUMMY	CHAR(1)																																																
DUMMY	CHAR(1)																																																		

Figure 15. Tables used at the Capture control server. The tables used by the Capture program, Apply program, and Capture triggers at the Capture control server. The columns that make up each table's unique index are listed in parentheses under the table name.

## Control tables used at the Capture control server (image 2 of 2)

<b>schema.IBMSNAP_PRUNE_SET</b> (TARGET_SERVER, APPLY_QUAL, SET_NAME) TARGET_SERVER CHAR(18) NOT NULL APPLY_QUAL CHAR(18) NOT NULL SET_NAME CHAR(18) NOT NULL SYNCHTIME TIMESTAMP SYNCHPOINT CHAR(10) FOR BIT DATA NOT NULL	<b>schema.IBMSNAP_REG_SYNCH</b> (TRIGGER_ME) TRIGGER_ME CHAR(1) NOT NULL
<b>schema.IBMSNAP_REGISTER</b> (SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL) SOURCE_OWNER VARCHAR(30) NOT NULL SOURCE_TABLE VARCHAR(128) NOT NULL SOURCE_VIEW_QUAL SMALLINT NOT NULL GLOBAL_RECORD CHAR(1) NOT NULL SOURCE_STRUCTURE SMALLINT NOT NULL SOURCE_CONDENSED CHAR(1) NOT NULL SOURCE_COMPLETE CHAR(1) NOT NULL CD_OWNER VARCHAR(30) CD_TABLE VARCHAR(128) PHYS_CHANGE_OWNER VARCHAR(30) PHYS_CHANGE_TABLE VARCHAR(128) CD_OLD_SYNCHPOINT CHAR(10) FOR BIT DATA CD_NEW_SYNCHPOINT CHAR(10) FOR BIT DATA DISABLE_REFRESH SMALLINT NOT NULL CCD_OWNER VARCHAR(30) CCD_TABLE VARCHAR(128) CCD_OLD_SYNCHPOINT CHAR(10) FOR BIT DATA SYNCHPOINT CHAR(10) FOR BIT DATA SYNCHTIME TIMESTAMP CCD_CONDENSED CHAR(1) CCD_COMPLETE CHAR(1) ARCH_LEVEL CHAR(4) NOT NULL DESCRIPTION CHAR(254) BEFORE_IMG_PREFIX VARCHAR(4) CONFLICT_LEVEL CHAR(1) CHG_UPD_TO_DEL_INS CHAR(1) CHGONLY CHAR(1) RECAPTURE CHAR(1) OPTION_FLAGS CHAR(4) NOT NULL STOP_ON_ERROR CHAR(1) STATE CHAR(1) STATE_INFO CHAR(8)	<b>schema.IBMSNAP_RESTART</b> (no unique index) MAX_COMMITSEQ CHAR(10) FOR BIT DATA NOT NULL MAX_COMMIT_TIME TIMESTAMP NOT NULL MIN_INFLIGHTSEQ CHAR(10) FOR BIT DATA NOT NULL CURR_COMMIT_TIME TIMESTAMP NOT NULL CAPTURE_FIRST_SEQ CHAR(10) FOR BIT DATA NOT NULL OS/400 <b>schema.IBMSNAP_RESTART</b> (JRN_LIB, JRN_NAME) MAX_COMMITSEQ CHAR(10) FOR BIT DATA NOT NULL MAX_COMMIT_TIME TIMESTAMP NOT NULL MIN_INFLIGHTSEQ CHAR(10) FOR BIT DATA NOT NULL CURR_COMMIT_TIME TIMESTAMP NOT NULL CAPTURE_FIRST_SEQ CHAR(10) FOR BIT DATA NOT NULL UID INTEGER NOT NULL SEQNR BIGINT NOT NULL JRN_LIB CHAR(10) NOT NULL JRN_NAME CHAR(10) NOT NULL STATUS CHAR(1)
OS/400 only <b>schema.IBMSNAP_REG_EXT</b> (VERSION, SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL) VERSION INT NOT NULL SOURCE_OWNER VARCHAR(30) NOT NULL SOURCE_TABLE VARCHAR(128) NOT NULL SOURCE_NAME CHAR(10) SOURCE_MBR CHAR(10) SOURCE_TABLE_RDB CHAR(18) JRN_LIB CHAR(10) JRN_NAME CHAR(10) FR_START_TIME TIMESTAMP SOURCE_VIEW_QUAL SMALLINT NOT NULL CMT_BEHAVIOR_CASE SMALLINT NOT NULL WITH DEFAULT MAX_ROWS_BTWN_CMTS SMALLINT NOT NULL WITH DEFAULT	<b>schema.IBMSNAP_SEQTABLE</b> (SEQ) SEQ INTEGER NOT NULL
	<b>schema.IBMSNAP_SIGNAL</b> (SIGNAL_TIME) SIGNAL_TIME TIMESTAMP NOT NULL WITH DEFAULT SIGNAL_TYPE VARCHAR(30) NOT NULL SIGNAL_SUBTYPE VARCHAR(30) SIGNAL_INPUT_IN VARCHAR(500) SIGNAL_STATE CHAR(1) NOT NULL SIGNAL_LSN CHAR(10) FOR BIT DATA
	<b>schema.IBMSNAP_UOW</b> (IBMSNAP_COMMITSEQ, IBMSNAP_LOGMARKER) IBMSNAP_UOWID CHAR(10) FOR BIT DATA NOT NULL IBMSNAP_COMMITSEQ CHAR(10) FOR BIT DATA NOT NULL IBMSNAP_LOGMARKER TIMESTAMP NOT NULL IBMSNAP_AUTHID VARCHAR(30) NOT NULL IBMSNAP_AUTHID VARCHAR(30) NOT NULL IBMSNAP_REJ_CODE CHAR(1) NOT NULL WITH DEFAULT IBMSNAP_APPLY_QUAL CHAR(18) NOT NULL WITH DEFAULT

Figure 16. Tables used at the Capture control server (continued). The tables used by the Capture program, Apply program, and Capture triggers at the Capture control server. The columns that make up each table's unique index are listed in parentheses under the table name.

## Control tables used at the Apply control server (image 1 of 2)

<b>ASN.IBMSNAP_APPLYTRAIL</b> (no unique index)	
APPLY_QUAL	CHAR(18) NOT NULL
SET_NAME	CHAR(18) NOT NULL
SET_TYPE	CHAR(1) NOT NULL
WHOS_ON_FIRST	CHAR(1) NOT NULL
ASNLOAD	CHAR(1)
FULL_REFRESH	CHAR(1)
EFFECTIVE_MEMBERS	INT
SET_INSERTED	INT NOT NULL
SET_DELETED	INT NOT NULL
SET_UPDATED	INT NOT NULL
SET_REWORKED	INT NOT NULL
SET_REJECTED_TRXS	INT NOT NULL
STATUS	SMALLINT NOT NULL
LASTRUN	TIMESTAMP NOT NULL
LASTSUCCESS	TIMESTAMP
SYNCHPOINT	CHAR(10) FOR BIT DATA
SYNCHTIME	TIMESTAMP
SOURCE_SERVER	CHAR (18) NOT NULL
SOURCE_ALIAS	CHAR(8)
SOURCE_OWNER	VARCHAR(30)
SOURCE_TABLE	VARCHAR(128)
SOURCE_VIEW_QUAL	SMALLINT
TARGET_SERVER	CHAR(18) NOT NULL
TARGET_ALIAS	CHAR(8)
TARGET_OWNER	VARCHAR(30) NOT NULL
TARGET_TABLE	VARCHAR(128) NOT NULL
CAPTURE_SCHEMA	VARCHAR(30) NOT NULL
TGT_CAPTURE_SCHEMA	VARCHAR(30)
FEDERATED_SRC_SRVR	VARCHAR(18)
FEDERATED_TGT_SRVR	VARCHAR(18)
JRN_LIB	CHAR(10)
JRN_NAME	CHAR(10)
COMMIT_COUNT	SMALLINT
OPTION_FLAGS	CHAR(4) NOT NULL
EVENT_NAME	CHAR(18)
ENDTIME	TIMESTAMP NOT NULL
SOURCE_CONN_TIME	WITH DEFAULT TIMESTAMP
SQLSTATE	CHAR(5)
SQLCODE	INT
SQLERRP	CHAR(8)
SQLERRM	VARCHAR(70)
APPERRM	VARCHAR(760)

<b>ASN.IBMSNAP_APPENQ</b> (APPLY_QUAL)	
APPLY_QUAL	CHAR(18)

<i>OS/400 only</i> <b>ASN.IBMSNAP_APPLY_JOB</b>	
APPLY_QUAL	CHAR(18) NOT NULL
CONTROL_SERVER	CHAR(18) NOT NULL
JOB_NAME	CHAR(10) NOT NULL
USER_NAME	CHAR(10) NOT NULL
JOB_NUMBER	CHAR(6) NOT NULL

<b>ASN.IBMSNAP_APPLYTRACE</b> (APPLY_QUAL, TRACE_TIME)	
APPLY_QUAL	CHAR(18) NOT NULL
TRACE_TIME	TIMESTAMP NOT NULL
OPERATION	CHAR(8) NOT NULL
DESCRIPTION	VARCHAR(1024) NOT NULL

Figure 17. Tables used at the Apply control server. The tables used by the Apply program at the Apply control server. The columns that make up each table's unique index are listed in parentheses under the table name.



## Control tables used at the Apply control server (image 2 of 2)

### ASN.IBMSNAP\_SUBS\_COLS

(APPLY\_QUAL, SET\_NAME, WHOS\_ON\_FIRST, TARGET\_OWNER, TARGET\_TABLE, TARGET\_NAME)

APPLY_QUAL	CHAR(18) NOT NULL
SET_NAME	CHAR(18) NOT NULL
WHOS_ON_FIRST	CHAR(1) NOT NULL
TARGET_OWNER	VARCHAR(30) NOT NULL
TARGET_TABLE	VARCHAR(128) NOT NULL
COL_TYPE	CHAR(1) NOT NULL
TARGET_NAME	VARCHAR(30) NOT NULL
IS_KEY	CHAR(1) NOT NULL
COLNO	SMALLINT NOT NULL
EXPRESSION	VARCHAR(254) NOT NULL

### ASN.IBMSNAP\_SUBS\_STMTS

(APPLY\_QUAL, SET\_NAME, WHOS\_ON\_FIRST, BEFORE\_OR\_AFTER, STMT\_NUMBER)

APPLY_QUAL	CHAR(18) NOT NULL
SET_NAME	CHAR(18) NOT NULL
WHOS_ON_FIRST	CHAR(1) NOT NULL
BEFORE_OR_AFTER	CHAR(1) NOT NULL
STMT_NUMBER	SMALLINT NOT NULL
EI_OR_CALL	CHAR(1) NOT NULL
SQL_STMT	VARCHAR(1024)
ACCEPT_SQLSTATES	VARCHAR(50)

### ASN.IBMSNAP\_SUBS\_EVENT

(EVENT\_NAME, EVENT\_TIME)

EVENT_NAME	CHAR(18) NOT NULL
EVENT_TIME	TIMESTAMP NOT NULL
END_SYNCHPOINT	CHAR(10) FOR BIT DATA
END_OF_PERIOD	TIMESTAMP

### ASN.IBMSNAP\_SUBS\_MEMBR

(APPLY\_QUAL, SET\_NAME, WHOS\_ON\_FIRST, SOURCE\_OWNER, SOURCE\_TABLE, SOURCE\_VIEW\_QUAL, TARGET\_OWNER, TARGET\_TABLE)

APPLY_QUAL	CHAR(18) NOT NULL
SET_NAME	CHAR(18) NOT NULL
WHOS_ON_FIRST	CHAR(1) NOT NULL
SOURCE_OWNER	VARCHAR(30) NOT NULL
SOURCE_TABLE	VARCHAR(128) NOT NULL
SOURCE_VIEW_QUAL	SMALLINT NOT NULL
TARGET_OWNER	VARCHAR(30) NOT NULL
TARGET_TABLE	VARCHAR(128) NOT NULL
TARGET_CONDENSED	CHAR(1) NOT NULL
TARGET_COMPLETE	CHAR(1) NOT NULL
TARGET_STRUCTURE	SMALLINT NOT NULL
PREDICATES	VARCHAR(1024)
MEMBER_STATE	CHAR(1)
TARGET_KEY_CHG	CHAR(1) NOT NULL
UOW_CD_PREDICATES	VARCHAR(1024)
JOIN_UOW_CD	CHAR(1)
LOADX_TYPE	SMALLINT
LOADX_SRC_N_OWNER	VARCHAR(30)
LOADX_SRC_N_TABLE	VARCHAR(128)

### ASN.IBMSNAP\_SUBS\_SET

(APPLY\_QUAL, SET\_NAME, WHOS\_ON\_FIRST)

APPLY_QUAL	CHAR(18) NOT NULL
SET_NAME	CHAR(18) NOT NULL
SET_TYPE	CHAR(1) NOT NULL
WHOS_ON_FIRST	CHAR(1) NOT NULL
ACTIVATE	SMALLINT NOT NULL
SOURCE_SERVER	CHAR(18) NOT NULL
SOURCE_ALIAS	CHAR(8)
TARGET_SERVER	CHAR(18) NOT NULL
TARGET_ALIAS	CHAR(8)
STATUS	SMALLINT NOT NULL
LASTRUN	TIMESTAMP NOT NULL
REFRESH_TYPE	CHAR(1) NOT NULL
SLEEP_MINUTES	INT
EVENT_NAME	CHAR(18)
LASTSUCCESS	TIMESTAMP
SYNCHPOINT	CHAR(10) FOR BIT DATA
SYNCHTIME	TIMESTAMP
CAPTURE_SCHEMA	VARCHAR(30) NOT NULL
TGT_CAPTURE_SCHEMA	VARCHAR(30)
FEDERATED_SRC_SRVR	VARCHAR(18)
FEDERATED_TGT_SRVR	VARCHAR(18)
JRN_LIB	CHAR(10)
JRN_NAME	CHAR(10)
OPTION_FLAGS	CHAR(4) NOT NULL
COMMIT_COUNT	SMALLINT
MAX_SYNCH_MINUTES	SMALLINT
AUX_STMTS	SMALLINT NOT NULL
ARCH_LEVEL	CHAR(4) NOT NULL

Figure 18. Tables used at the Apply control server (continued). The tables used by the Apply program at the Apply control server. The columns that make up each table's unique index are listed in parentheses under the table name.

## Control tables used at the Monitor control server

<b>ASN.IBMSNAP_ALERTS</b> (MONITOR_QUAL, COMPONENT, SERVER_NAME, SCHEMA_OR_QUAL, CONDITION_NAME, ALERT_CODE)  MONITOR_QUAL CHAR(18) NOT NULL ALERT_TIME TIMESTAMP NOT NULL COMPONENT CHAR(1) NOT NULL SERVER_NAME CHAR(18) NOT NULL SERVER_ALIAS CHAR(8) SCHEMA_OR_QUAL VARCHAR(30) NOT NULL SET_NAME CHAR(18) NOT NULL WITH DEFAULT CONDITION_NAME CHAR(18) NOT NULL OCCURRED_TIME TIMESTAMP NOT NULL ALERT_COUNTER SMALLINT NOT NULL ALERT_CODE CHAR(10) NOT NULL RETURN_CODE INT NOT NULL NOTIFICATION_SENT CHAR(1) NOT NULL ALERT_MESSAGE VARCHAR(1024) NOT NULL	<b>ASN.IBMSNAP_GROUPS</b> (GROUP_NAME) GROUP_NAME VARCHAR(127) NOT NULL DESCRIPTION VARCHAR(1024)
<b>ASN.IBMSNAP_CONDITIONS</b> (MONITOR_QUAL, SERVER_NAME, COMPONENT, SCHEMA_OR_QUAL, SET_NAME, CONDITION_NAME)  SERVER_NAME CHAR(18) NOT NULL COMPONENT CHAR(1) NOT NULL SCHEMA_OR_QUAL VARCHAR(30) NOT NULL SET_NAME CHAR(18) NOT NULL WITH DEFAULT MONITOR_QUAL CHAR(18) NOT NULL SERVER_ALIAS CHAR(8) ENABLED CHAR(1) NOT NULL CONDITION_NAME CHAR(18) NOT NULL PARAM_INT INT PARAM_CHAR VARCHAR(128) CONTACT_TYPE CHAR(1) NOT NULL CONTACT VARCHAR(127) NOT NULL	<b>ASN.IBMSNAP_MONENQ</b> (MONITOR_QUAL)  MONITOR_QUAL CHAR(18) NOT NULL
<b>ASN.IBMSNAP_CONTACTGRP</b> (GROUP_NAME, CONTACT_NAME)  GROUP_NAME VARCHAR(127) NOT NULL CONTACT_NAME VARCHAR(127) NOT NULL	<b>ASN.IBMSNAP_MONSERVERS</b> (MONITOR_QUAL, SERVER_NAME)  MONITOR_QUAL CHAR(18) NOT NULL SERVER_NAME CHAR(18) NOT NULL SERVER_ALIAS CHAR(8) LAST_MONITOR_TIME TIMESTAMP NOT NULL START_MONITOR_TIME TIMESTAMP END_MONITOR_TIME TIMESTAMP LASTRUN TIMESTAMP NOT NULL LASTSUCCESS TIMESTAMP STATUS SMALLINT NOT NULL
<b>ASN.IBMSNAP_CONTACTS</b> (CONTACT_NAME)  CONTACT_NAME VARCHAR(127) NOT NULL EMAIL_ADDRESS VARCHAR(128) NOT NULL ADDRESS_TYPE CHAR(1) NOT NULL DELEGATE VARCHAR(127) DELEGATE_START DATE DELEGATE_END DATE DESCRIPTION VARCHAR(1024)	<b>ASN.IBMSNAP_MONTRACE</b> (MONITOR_QUAL, TRACE_TIME)  MONITOR_QUAL CHAR(18) NOT NULL TRACE_TIME TIMESTAMP NOT NULL OPERATION CHAR(8) NOT NULL DESCRIPTION VARCHAR(1024) NOT NULL
<b>ASN.IBMSNAP_CONTACTS</b> (CONTACT_NAME)  CONTACT_NAME VARCHAR(127) NOT NULL EMAIL_ADDRESS VARCHAR(128) NOT NULL ADDRESS_TYPE CHAR(1) NOT NULL DELEGATE VARCHAR(127) DELEGATE_START DATE DELEGATE_END DATE DESCRIPTION VARCHAR(1024)	<b>ASN.IBMSNAP_MONTRAIL</b> (no unique index)  MONITOR_QUAL CHAR(18) NOT NULL SERVER_NAME CHAR(18) NOT NULL SERVER_ALIAS CHAR(8) STATUS SMALLINT NOT NULL LASTRUN TIMESTAMP NOT NULL LASTSUCCESS TIMESTAMP ENDTIME TIMESTAMP NOT NULL WITH DEFAULT LAST_MONITOR_TIME TIMESTAMP NOT NULL START_MONITOR_TIME TIMESTAMP END_MONITOR_TIME TIMESTAMP SQLCODE INT SQLSTATE CHAR(5) NUM_ALERTS INT NOT NULL NUM_NOTIFICATIONS INT NOT NULL

Figure 19. Tables used at the Monitor control server. The tables used by the Replication Alert Monitor program at the Monitor control server. The columns that make up each table's unique index are listed in parentheses under the table name.

## List of tables used at the Capture control server

The tables stored at the Capture control server contain information about your registered sources and how the Capture program or triggers process the sources. For UNIX, Windows, and z/OS, you build these control tables to your specifications using the Replication Center. For OS/400, these control tables are created automatically for you in the ASN library when you install DataPropagator for iSeries. You can use the system commands for replication on OS/400 to create Capture control tables in alternate capture schemas.

Table 52. Quick reference for tables used at the Capture control server

Table name	Description	See page
ASN.IBMSNAP_CAPSCHEMAS	<b>Capture schemas table</b>  Contains the names of all Capture schemas.	483
<i>schema</i> .IBMSNAP_AUTHTKN (OS/400)	<b>Apply-qualifier cross-reference table</b>  Contains information to support update-anywhere.	484
<i>schema</i> .IBMSNAP_CAPENQ (UNIX, Windows, z/OS)	<b>Capture enqueue table</b>  For each Capture schema, this table is used to ensure that: <ul style="list-style-type: none"> <li>• For DB2 for UNIX and Windows, only one Capture program is running per database.</li> <li>• For non-data-sharing DB2 for z/OS, only one Capture program is running per subsystem.</li> <li>• For data-sharing DB2 for z/OS, only one Capture program is running per data-sharing group.</li> </ul>	485
<i>schema</i> .CD_table	<b>Change-data (CD) table</b>  Contains information about changes that occur at the source. This table is not created until you register a replication source.	493
<i>schema</i> .CCD_table	<b>Consistent-change-data (CCD) table</b>  Contains information about changes that occur at the source and additional columns to identify the sequential ordering of those changes.	491
<i>schema</i> .IBMSNAP_CAPMON	<b>Capture monitor table</b>  Contains operational statistics that help monitor the progress of the Capture program.	486
<i>schema</i> .IBMSNAP_CAPPARMS	<b>Capture parameters table</b>  Contains parameters that you can specify to control the operations of the Capture program.	487
<i>schema</i> .IBMSNAP_CAPTRACE	<b>Capture trace table</b>  Contains important messages from the Capture program.	490
<i>schema</i> .IBMSNAP_PRUNE_LOCK	<b>Prune lock table</b>  Used to serialize the Capture program's access of CD tables during a cold start or during retention-limit pruning.	496

Table 52. Quick reference for tables used at the Capture control server (continued)

Table name	Description	See page
<i>schema</i> .IBMSNAP_PRUNE_SET	<b>Prune set table</b>  Coordinates the pruning of CD tables.	496
<i>schema</i> .IBMSNAP_PRUNCNTL	<b>Pruning control table</b>  Coordinates synchpoint updates between the Capture and Apply programs.	494
<i>schema</i> .IBMSNAP_REG_EXT (OS/400)	<b>Register extension table</b>  An extension of the register table. Contains additional information about replication sources, such as the journal name and the remote source table's database entry name.	497
<i>schema</i> .IBMSNAP_REGISTER	<b>Register table</b>  Contains information about replication sources, such as the names of the replication source tables, their attributes, and their corresponding CD and CCD table names.	498
<i>schema</i> .IBMSNAP_REG_SYNCH (non-DB2 relational)	<b>Register synchronization table</b>  Used when replicating from a non-DB2 relational data source. An update trigger on this table simulates the Capture program by initiating an update of the SYNCHPOINT value for all the rows in the register table before the Apply program reads the information from the register table.	505
<i>schema</i> .IBMSNAP_RESTART	<b>Restart table</b>  Contains information that enables the Capture program to resume capturing from the correct point in the log or journal. For OS/400 environments, this table is also used to determine the starting time of the <b>RCVJRNE</b> (Receive Journal Entry) command.	505
<i>schema</i> .IBMSNAP_SEQTABLE (non-DB2 relational)	<b>Sequencing table</b>  Contains a sequence of unique numbers that DB2 replication uses as the equivalent of log sequence numbers for Informix tables.	507
<i>schema</i> .IBMSNAP_SIGNAL	<b>Signal table</b>  Contains all signals used to prompt the Capture program. These signals can be sent manually or by the Apply program.	507

Table 52. Quick reference for tables used at the Capture control server (continued)

Table name	Description	See page
<i>schema</i> .IBMSNAP_UOW	<b>Unit-of-work (UOW) table</b>  Provides additional information about transactions that have been committed to a source table.	510

**Related reference:**

- “ASN.IBMSNAP\_CAPSCHEMAS” on page 483
- “*schema*.IBMSNAP\_AUTHTKN (OS/400)” on page 484
- “*schema*.IBMSNAP\_CAPENQ (UNIX, Windows, z/OS)” on page 485
- “*schema*.IBMSNAP\_CAPMON” on page 486
- “*schema*.IBMSNAP\_CAPPARMS” on page 487
- “*schema*.IBMSNAP\_CAPTRACE (DB2 only)” on page 490
- “*schema*.CCD\_table (non-DB2)” on page 491
- “*schema*.CD\_table” on page 493
- “*schema*.IBMSNAP\_PRUNCNTL” on page 494
- “*schema*.IBMSNAP\_PRUNE\_LOCK” on page 496
- “*schema*.IBMSNAP\_PRUNE\_SET” on page 496
- “*schema*.IBMSNAP\_REG\_EXT (OS/400)” on page 497
- “*schema*.IBMSNAP\_REGISTER” on page 498
- “*schema*.IBMSNAP\_REG\_SYNCH (non-DB2 relational)” on page 505
- “*schema*.IBMSNAP\_RESTART” on page 505
- “*schema*.IBMSNAP\_SEQTABLE (Informix)” on page 507
- “*schema*.IBMSNAP\_SIGNAL” on page 507
- “*schema*.IBMSNAP\_UOW” on page 510

## List of tables used at the Apply control server

The tables stored at the Apply control server contain information about your subscription definitions. For UNIX, Windows, and z/OS, you build these control tables to your specifications using the Replication Center. For OS/400, these control tables are created automatically for you when install DataPropagator for iSeries.

Table 53. Quick reference for tables used at the Apply control server

Table name	Description	See page
ASN.IBMSNAP_APPENQ	<b>Apply enqueue table</b>  Used to ensure that only one Apply program is running per Apply qualifier.	513
ASN.IBMSNAP_APPLY_JOB (OS/400)	<b>Apply job table</b>  Used to ensure that there is a unique Apply qualifier for each instance of the Apply program running at an Apply control server.	513
ASN.IBMSNAP_APPLYTRACE	<b>Apply trace table</b>  Contains important messages from the Apply program.	514
ASN.IBMSNAP_APPLYTRAIL	<b>Apply trail table</b>  Contains audit-trail information about the Apply program.	515
ASN.IBMSNAP_SUBS_COLS	<b>Subscription columns table</b>  Maps columns in the target table or view to the corresponding columns in the source table or view.	519
ASN.IBMSNAP_SUBS_EVENT	<b>Subscription events table</b>  Contains events that you define to control when the Apply program processes a subscription set.	521
ASN.IBMSNAP_SUBS_MEMBR	<b>Subscription members table</b>  Identifies a source and target table pair and specifies processing information for that pair.	522
ASN.IBMSNAP_SUBS_SET	<b>Subscription sets table</b>  Contains processing information for each set of subscription-set members that the Apply program processes as a group.	526
ASN.IBMSNAP_SUBS_STMTS	<b>Subscription statements table</b>  Contains SQL statements or stored procedure calls that you define for a subscription set. They are invoked before or after the Apply program processes the set.	531

**Related reference:**

- “ASN.IBMSNAP\_APPENQ” on page 513
- “ASN.IBMSNAP\_APPLY\_JOB (OS/400)” on page 513
- “ASN.IBMSNAP\_APPLYTRACE” on page 514

- “ASN.IBMSNAP\_APPLYTRAIL” on page 515
- “ASN.IBMSNAP\_SUBS\_COLS” on page 519
- “ASN.IBMSNAP\_SUBS\_EVENT” on page 521
- “ASN.IBMSNAP\_SUBS\_MEMBR” on page 522
- “ASN.IBMSNAP\_SUBS\_SET” on page 526
- “ASN.IBMSNAP\_SUBS\_STMTS” on page 531

---

## List of tables used at the Monitor control server

The tables at the Monitor control server contain information about when, how, and whom you want the Replication Alert Monitor to contact when an alert condition occurs. For UNIX, Windows, and z/OS, you build these control tables to your specifications using the Replication Center. DataPropagator for iSeries does not have Monitor control tables.

*Table 54. Quick reference for tables used at the Monitor control server*

Table name	Description	See page
ASN.IBMSNAP_ALERTS	<b>Monitor alerts table</b>  Contains a record of all the alerts issued by the Replication Alert Monitor.	533
ASN.IBMSNAP_CONDITIONS	<b>Monitor conditions table</b>  Contains the alert conditions for which the Replication Alert Monitor will contact someone, and contains the group or individual's name to contact if a particular condition occurs.	534
ASN.IBMSNAP_CONTACTGRP	<b>Monitor group contacts table</b>  Contains the individual contacts that make up the contact groups.	537
ASN.IBMSNAP_CONTACTS	<b>Monitor contacts table</b>  Contains information on how the Replication Alert Monitor notifies each person or group when an alert condition that is associated with that contact name occurs.	537
ASN.IBMSNAP_GROUPS	<b>Monitor groups table</b>  Contains the name and description of each contact group.	538
ASN.IBMSNAP_MONENQ	<b>Monitor enqueue table</b>  Used to ensure that only one Replication Alert Monitor program is running per Monitor qualifier.	538

---

Table 54. Quick reference for tables used at the Monitor control server (continued)

Table name	Description	See page
ASN.IBMSNAP_MONSERVERS	<b>Monitored servers table</b>  Contains the latest time that a server was monitored by a Replication Alert Monitor program (identified by a Monitor qualifier).	539
ASN.IBMSNAP_MONTRACE	<b>Monitor trace table</b>  Contains important messages from the Monitor program.	540
ASN.IBMSNAP_MONTRAIL	<b>Monitor trail table</b>  Contains important information about each monitor cycle.	540

**Related reference:**

- “ASN.IBMSNAP\_ALERTS” on page 533
- “ASN.IBMSNAP\_CONDITIONS” on page 534
- “ASN.IBMSNAP\_CONTACTGRP” on page 537
- “ASN.IBMSNAP\_MONTRAIL” on page 540
- “ASN.IBMSNAP\_CONTACTS” on page 537
- “ASN.IBMSNAP\_GROUPS” on page 538
- “ASN.IBMSNAP\_MONENQ” on page 538
- “ASN.IBMSNAP\_MONSERVERS” on page 539
- “ASN.IBMSNAP\_MONTRACE” on page 540

## List of tables used at the target server

Various types of target tables are stored at the target server. If you do not use an existing table as your target table, the Replication Center builds the target table to your specifications based on how you define the subscription-set member.

Table 55. Quick reference for target tables

Table name	Description	See page
schema.base_aggregate	<b>Base aggregate table</b>  Contains data that has been aggregated from a source table.	541
schema.change_aggregate	<b>Change aggregate table</b>  Contains data that has been aggregated from a CD table.	542



Table 55. Quick reference for target tables (continued)

Table name	Description	See page
<i>schema.CCD</i>	<b>Consistent-change data (CCD) table</b>  Contains information about changes that occur at the source and contains additional columns to identify the sequential ordering of those changes.	543
<i>schema.point_in_time</i>	<b>Point-in-time table</b>  A copy of the source data, with an additional column that records the specific time in the source log that the data was committed.	546
<i>schema.replica</i>	<b>Replica table</b>  A type of target table used for update-anywhere replication.	546
<i>schema.user_copy</i>	<b>User copy table</b>  A copy of the source table.	547

**Related reference:**

- “Base aggregate table” on page 541
- “Change aggregate table” on page 542
- “Consistent-change data (CCD) table” on page 543
- “Point-in-time table” on page 546
- “Replica table” on page 546
- “User copy table” on page 547

## Tables at the Capture control server and their column descriptions

This section provides greater detail of each table stored at the Capture control server. It also lists and briefly describes the columns in each table. The control tables are listed in alphabetical order, and the columns are listed in the order that they appear in each table from left to right.

### ASN.IBMSNAP\_CAPSCHEMAS

**Server:** Capture control server

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results while using the administration tools.

ASN.IBMSNAP\_CAPSCHEMAS

The Capture schemas table holds the names of all Capture schemas. It allows the Replication Center and other utilities to quickly find all of the tables for a given Capture control server. A row is automatically inserted each time you create a new Capture schema.

The following two tables show operating system-specific layouts of the Capture schemas table:

Table 56. Columns in the Capture schemas table for all operating systems other than OS/400

Column name	Description
CAP_SCHEMA_NAME	The name of a Capture schema. A row exists for each Capture schema.

Table 57. Columns in the Capture schemas table for OS/400

Column name	Description
CAP_SCHEMA_NAME	The name of a Capture schema. A row exists for each Capture schema.
STATUS	A flag that indicates whether the Capture program that is identified by this Capture schema is running:  Y        The Capture program is running.  N        The Capture program is not running.

schema.IBMSNAP\_AUTHTKN (OS/400)

**Server:** Capture control server

**Default schema:** ASN

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The Apply-qualifier cross-reference table is used in the OS/400 environment only. This table is used during update-anywhere replication to keep track of the transactions that have been processed by a particular Apply program, which is identified by an Apply qualifier. The Capture program prunes this table based on the retention limit that you set.

Table 58 on page 485 provides a brief description of the columns in the Apply-qualifier cross-reference table.

Table 58. Columns in the apply-qualifier cross-reference table

Column name	Description
APPLY_QUAL	The Apply qualifier that identifies which Apply program processed the transaction. This qualifier is used during update-anywhere replication to prevent the Apply program from replicating the same changes repeatedly.
IBMSNAP_AUTHTKN	The job name that is associated with the transaction. Capture for iSeries matches the name in this column with the name of the job that issued the transaction to determine whether the transaction was issued by either the Apply program or a user application. If the job names match, then Capture for iSeries copies the Apply qualifier that's in the APPLY_QUAL column of this table to the APPLY_QUAL column in the corresponding row of the UOW table. If the names do not match, then Capture for iSeries sets the APPLY_QUAL column of the UOW row to null. This column is not automatically copied to other tables; you must select it and copy it as a user data column.
JRN_LIB	The library name of the journal from which the transactions came.
JRN_NAME	The name of the journal from which the transactions came.
IBMSNAP_LOGMARKER	The approximate time that the transaction was committed at the Capture control server.

### schema.IBMSNAP\_CAPENQ (UNIX, Windows, z/OS)

**Server:** Capture control server

**Default schema:** ASN

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The Capture enqueue table is not used on non-DB2 relational or OS/400 servers.

For a single Capture schema, the Capture enqueue table ensures that:

- For DB2 for the workstation, only one Capture program is running per database
- For non-data-sharing DB2 for z/OS, only one Capture program is running per subsystem
- For data-sharing DB2 for z/OS, only one Capture program is running per data-sharing group

While running, the Capture program exclusively locks this table.

Table 59 on page 486 provides a brief description of the column in the Capture enqueue table.

Table 59. Column in the Capture enqueue table

Column name	Description
LOCKNAME	This column contains no data.

schema.**IBMSNAP\_CAPMON**

**Server:** Capture control server

**Default schema:** ASN

The Capture program inserts a row in the Capture monitor table after each interval to provide you with operational statistics. The Replication Center uses information in this table (and in other tables) so you can monitor the status of the Capture program. In the Capture parameters (IBMSNAP\_CAPPARMS) table, the value that you specify for MONITOR\_INTERVAL indicates how frequently the Capture program makes inserts into the Capture monitor table, and the value that you specify for the MONITOR\_LIMIT indicates the number of minutes that rows remain in the table before they are eligible for pruning.

Table 60 provides a brief description of the columns in the Capture monitor table.

Table 60. Columns in the Capture monitor table

Column name	Description
MONITOR_TIME	The timestamp (at the Capture control server) when the row was inserted into this table.
RESTART_TIME	The timestamp when the current invocation of the Capture program was restarted.
CURRENT_MEMORY	The amount of memory (in megabytes) that the Capture program used.
CD_ROWS_INSERTED	The number of rows that the Capture program inserted into the CD table for all source tables.
RECAP_ROWS_SKIPPED	For update-anywhere replication, this is the number of rows that the Capture program processed but did not insert into the CD table. The rows were skipped because the registration was defined for the Capture program to not recapture changes that have been replicated to this table that did not originate at this source server.
TRIGR_ROWS_SKIPPED	The number of rows that the Capture program processed but did not insert into the CD table. The rows were skipped because you defined a trigger on the registration for the Capture program to suppress certain rows.
CHG_ROWS_SKIPPED	The number of rows that the Capture program processed but did not insert into the CD table. The rows were skipped because the registration was defined for the Capture program to only capture changes that occur in registered columns.

Table 60. Columns in the Capture monitor table (continued)

Column name	Description
TRANS_PROCESSED	The number of transactions at the source system that the Capture program processed.
TRANS_SPILLED	The number of transactions at the source system that the Capture program spilled to disk due to memory restrictions.
MAX_TRAN_SIZE	The largest transaction that occurred at the source system. Knowing the transaction size might influence you to change the memory parameters.
LOCKING_RETRIES	The number of times a deadlock caused rework.
JRN_LIB (OS/400)	The library name of the journal that the Capture program was processing.
JRN_NAME (OS/400)	The name of the journal that the Capture program was processing.
LOGREADLIMIT	A count of the number of times log reading was interrupted before completing the building of a transaction because it reached an internal limit of the number of log records to be read.
CAPTURE_IDLE	The number of times that the Capture program slept because it didn't have any work to process.
SYNCHTIME	The current value of SYNCHTIME read from the global row of the register table when the monitor record was inserted into this table.

**schema.IBMSNAP\_CAPPARMS****Server:** Capture control server**Default schema:** ASN

This table contains information that you can update by using SQL.

The Capture parameters table contains parameters that you can modify to control the operations of the Capture program. You can define these parameters to set things such as the length of time that the Capture program retains data in the CD and UOW tables before pruning and the amount of time that the Capture program is allowed to lag in processing log records. If you make changes to the parameters in this table, the Capture program reads your modifications only during startup.

Table 61 on page 488 provides a brief description of the columns in the Capture parameters table.

Table 61. Columns in the Capture parameters table

Column name	Description
RETENTION_LIMIT	The length of time that rows remain in the CD, UOW, and signal tables before they become eligible for pruning, in cases where they have not been pruned based on the normal criteria. Normally, CD and UOW rows are pruned after they are applied to all targets, and signal rows are pruned when their cycle is complete (SIGNAL_STATE = C).
LAG_LIMIT	The number of minutes that the Capture program is allowed to lag when processing log records before it shuts itself down. During periods of high update frequency, full refreshes can be more economical than updates.
COMMIT_INTERVAL	How often, in seconds, the Capture program commits data to the Capture control tables, including the UOW and CD tables. This value should be less than the DB2 lockout value to prevent contention between the Capture and pruning threads.
PRUNE_INTERVAL	How often, in seconds, the Capture program automatically prunes (AUTOPRUNE = Y) rows in the CD, UOW, signal, trace, and Capture monitor tables that are no longer needed. A lower prune interval saves space, but increases processing costs. A higher prune interval requires more CD and UOW table space, but decreases processing costs.
TRACE_LIMIT	The number of minutes that rows remain in the Capture trace (IBMSNAP_CAPTRACE) table before they are eligible for pruning. During the pruning process, the rows in the Capture trace table are pruned if the number of minutes (current timestamp – the time a row was inserted in the Capture trace table) exceeds the value of TRACE_LIMIT.
MONITOR_LIMIT	The number of minutes that rows remain in the Capture monitor (IBMSNAP_CAPMON) table before they are eligible for pruning. During the pruning process, rows in the Capture monitor table are pruned if the value of minutes (current timestamp – MONITOR_TIME) exceeds the value of MONITOR_LIMIT.
MONITOR_INTERVAL	How often, in seconds, that the monitor thread adds a row to the Capture monitor (IBMSNAP_CAPMON) table. For Capture for iSeries, enter an interval greater than 120 seconds.
MEMORY_LIMIT	The amount of memory, in megabytes, that the Capture program is allowed to use. After this allocation is used up, memory transactions will spill to a file.
REMOTE_SRC_SERVER	Reserved for future options of DB2 replication. Currently this column contains the default value of null.

Table 61. Columns in the Capture parameters table (continued)

Column name	Description
AUTOPRUNE	<p>A flag that indicates whether the Capture program automatically prunes rows that are no longer needed from the CD, UOW, signal, trace, and Capture monitor tables:</p> <p><b>Y</b>      Autopruning is on.</p> <p><b>N</b>      Autopruning is off.</p>
TERM	<p>A flag that indicates whether the Capture program terminates when DB2 quiesces:</p> <p><b>Y</b>      The Capture program terminates when DB2 terminates.</p> <p><b>N</b>      The Capture program stays active and waits for DB2 to be restarted.</p>
AUTOSTOP	<p>A flag that indicates whether the Capture program stops capturing changes as soon as it reaches the end of the active logs:</p> <p><b>Y</b>      The Capture program stops as soon as it reaches the end of the active logs.</p> <p><b>N</b>      The Capture program continues running when it reaches the end of the active logs.</p>
LOGREUSE	<p>A flag that indicates whether the Capture program overwrites the Capture log file or appends to it.</p> <p><b>Y</b>      The Capture program reuses the log file by first deleting it and then recreating it when the Capture program is restarted.</p> <p><b>N</b>      The Capture program appends new information to the Capture log file.</p>
LOGSTDOUT	<p>A flag that indicates where the Capture program directs the log file messages:</p> <p><b>Y</b>      The Capture program directs log file messages to both the standard out (STDOUT) and the log file.</p> <p><b>N</b>      The Capture program directs most log file messages to the log file only. Initialization messages go to both the standard out (STDOUT) and the log file.</p>
SLEEP_INTERVAL (UNIX, Windows, z/OS)	The number of seconds that the Capture program sleeps when it reaches the end of the active logs (in UNIX and Windows, or in z/OS non-data-sharing environments), or when an inefficient amount of data has been returned (in z/OS data-sharing environments).
CAPTURE_PATH	The path where the output from the Capture program is sent.

Table 61. Columns in the Capture parameters table (continued)

Column name	Description
STARTMODE	<p>The processing procedure that the Capture program uses when it is started:</p> <p><b>cold</b> The Capture program deletes all rows in its CD tables and UOW table during initialization. All subscriptions to these replication sources are fully refreshed during the next Apply processing cycle (that is, all data is copied from the source tables to the target tables). If the Capture program tries to cold start but you disabled full refresh, the Capture program will start but the Apply program will fail and will issue an error message.</p> <p><b>warmsi</b> The Capture program warm starts; except if this is the first time you are starting the Capture program then it switches to a cold start. The warmsi start mode ensures that cold starts happen only when you initially start the Capture program.</p> <p><b>warmns</b> The Capture program warm starts. If it can't warm start, it does <i>not</i> switch to cold start. The warmns start mode prevents cold starts from occurring unexpectedly and is useful when problems arise (such as unavailable databases or table spaces) that require repair and that prevent a warm start from proceeding. When the Capture program warm starts, it resumes processing where it ended. If errors occur after the Capture program started, the Capture program terminates and leaves all tables intact.</p> <p><b>warmsa</b> If warm start information is available, the Capture program resumes processing where it ended in its previous run. If the Capture program cannot warm start, it switches to a cold start that refreshes all of your target tables.</p>

schema.IBMSNAP\_CAPTRACE (DB2 only)

Server: Capture control server

Default schema: ASN

The Capture trace table contains important messages from the Capture program.

The following two tables show operating system-specific layouts of the Capture trace table.



Table 62. Columns in the Capture trace table for UNIX, Windows, and z/OS

Column name	Description
OPERATION	The type of Capture program operation, for example, initialization, capture, or error condition.
TRACE_TIME	The time at the Capture control server that the row was inserted in the Capture trace table.
DESCRIPTION	The message ID followed by the message text. It can be an error message, a warning message, or an informational message. This column contains English-only text.

Table 63. Columns in the Capture trace table for OS/400

Column name	Description
OPERATION	The type of operation that the Capture program performed, for example, initialization, capture, or error condition.
TRACE_TIME	The time that the row was inserted in the Capture trace table. TRACE_TIME rows that are eligible for trace limit pruning will be deleted when the Capture program prunes the CD and UOW tables.
JOB_NAME	The fully qualified name of the job that wrote this trace entry.
	<b>Position</b> <b>Description</b> <b>1–10</b> QDPRCTL5 or the journal job name <b>11–20</b> The ID of the user who started the Capture program <b>21–26</b> The job number
JOB_STR_TIME	The starting time of the job that is named in the JOB_NAME column.
DESCRIPTION	The message ID followed by the message text. The message ID is the first seven characters of the DESCRIPTION column. The message text starts at the ninth position of the DESCRIPTION column.

### schema.CCD\_table (non-DB2)

**Server:** Capture control server

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause a loss of data.

Consistent-change-data (CCD) tables at the Capture control server are tables that contain information about changes that occur at a non-DB2 source and additional columns to identify the sequential ordering of those changes. A CCD table at the Capture control server is a table that is populated by a program other than the Apply program. It can be either:

- An internal CCD table for a non-DB2 relational source.

For change-capture replication, the Capture triggers insert changes in this table as updates occur at the non-DB2 relational source. The name of this type of CCD table is stored on the same row in the register (IBMSNAP\_REGISTER) table as the replication source that it holds changes from. This table is automatically pruned by the pruning trigger that is created when you register a non-DB2 relational source.

- An external CCD table for non-relational and multi-vendor data.  
External programs can create CCD tables to be used by DB2 replication as replication sources. These external programs capture IMS changes in a CCD table, so that copies of IMS data can be recreated in a relational database. The external programs must initialize, maintain, and supply the correct values for the control columns. If you have externally populated CCD tables that are not maintained by a program such as IMS DataPropagator or DataRefresher, you must maintain these tables yourself so that the Apply program can read the CCD tables as sources and function correctly. For details on how to maintain an externally populated CCD, see “Maintaining CCD tables as sources (IMS)” on page 61.

For information on CCD tables as targets in a subscription-set member, see “Consistent-change data (CCD) table” on page 543.

Table 64 provides a brief description of the columns in the CCD table.

Table 64. Columns in the CCD table

Column name	Description
IBMSNAP_INTENTSEQ	A sequence number that uniquely identifies a change. This value is globally ascending.
IBMSNAP_OPERATION	A flag that indicates the type of operation for a record:  <b>I</b> Insert  <b>U</b> Update  <b>D</b> Delete
IBMSNAP_COMMITSEQ	A sequence number that provides transactional order.
IBMSNAP_LOGMARKER	The time that the data was committed.
<i>user key columns</i>	If the CCD table is condensed, this column contains the columns that make up the target key.
<i>user non-key columns</i>	The non-key data columns from the source table. The column names that are in the source table do not need to match these column names, but the data types must be compatible.
<i>user computed columns</i>	User-defined columns that are derived from SQL expressions. You can use computed columns with SQL functions to convert source data types to different target data types.

**Related reference:**

- “Consistent-change data (CCD) table” on page 543

*schema.CD\_table*

**Server:** Capture control server

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause a loss of data.

Change-data (CD) tables record all committed changes made to a replication source. Pruning of the CD table is coordinated by the prune set (IBMSNAP\_PRUNE\_SET) table. (See “*schema.IBMSNAP\_PRUNE\_SET*” on page 496 for more information about how the CD table gets pruned.) Unlike other Capture control tables, CD tables are created when you define a replication source; they are not created automatically when you generate the control tables for the Capture control server.

Table 65 provides a list and a brief description of the columns in the CD table.

*Table 65. Columns in the CD table*

Column name	Description
IBMSNAP_COMMITSEQ	The log sequence number of the captured commit statement. This column, which is also in the UOW table, is included in the CD table to allow the Apply program to process user copy target tables without needing to join the CD table with the UOW table. In cases where a join between the CD table and the UOW table is required, the join is done using the IBMSNAP_COMMITSEQ column.
IBMSNAP_INTENTSEQ	The log sequence number of the log record of the change (insert, update, or delete). This value is globally ascending. If you selected for updates to be processes as delete/insert pairs, the IBMSNAP_INTENTSEQ value for the delete row is manufactured to be slightly smaller than the corresponding value for the insert row.
IBMSNAP_OPERATION	A flag that indicates the type of operation for a record:  <div> <b>I</b>      Insert  <b>U</b>      Update  <b>D</b>      Delete </div>
<i>user column after-image</i>	In most cases, the after-image column contains the value that is in the source column after the change occurs. This column has the same name, data type, and null attributes as the source column. In the case of an update, this column reflects the new value of the data that was updated. In the case of a delete, this column reflects the value of the data that was deleted. In the case of an insert, this column reflects the value of the data that was inserted.

Table 65. Columns in the CD table (continued)

Column name	Description
user column before-image	This column only exists in the CD table if you registered the source to include before-image column values. In most cases, the before-image column contains the value that was in the source column before the change occurred. This column has the same name as the source column, prefixed by the value in the BEFORE_IMG_PREFIX column in the register (IBMSNAP_REGISTER) table. It also has the same data type as the source column; however, it always allows null values for insert operations regardless of the source column's null attributes. In the case of an update, this column reflects the data that was updated. In the case of a delete, this column reflects the data that was deleted. In the case of an insert, this column is null.

schema.IBMSNAP\_PRUNCNTL  
schema.IBMSNAP\_PRUNCNTL

Server: Capture control server

Default schema: ASN

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The pruning control table contains detailed information regarding all subscription set members that are defined for this Capture schema. This table is used in conjunction with the prune set (IBMSNAP\_PRUNE\_SET) table during pruning. It is also used during the initialization handshake process between the Apply and Capture programs.

For DB2 sources, you can invoke pruning by issuing the **prune** command or have it done automatically. See “*schema.IBMSNAP\_CAPPARMS*” on page 487 for more information about using the Capture parameters table to set AUTOPRUNE. For non-DB2 relational sources, pruning is done by the pruning trigger that was created when you registered the source.

Table 66 provides a brief description of the columns in the pruning control table.

Table 66. Columns in the pruning control table

Column name	Description
TARGET_SERVER	The server name where target table or view for this member resides.
TARGET_OWNER	The high-level qualifier for the target table or view for this member.
TARGET_TABLE	The name of the target table or view for this member.

Table 66. Columns in the pruning control table (continued)

Column name	Description														
SYNCHTIME	The Capture program sets this timestamp during the initialization handshake process with the Apply program. The value comes from the timestamp of the commit log record that is associated with the transaction of the CAPSTART signal insert. It will not be updated again unless a subsequent initialization process takes place.														
SYNCHPOINT	The Capture program sets this value during the initialization handshake process with the Apply program. The value comes from the log sequence number of the commit log record that is associated with the transaction of the CAPSTART signal insert. It will not be updated again unless a subsequent initialization process takes place.														
SOURCE_OWNER	The high-level qualifier of the source table or view for this member.														
SOURCE_TABLE	The name of the source table or view for this member.														
SOURCE_VIEW_QUAL	This column is used to support multiple registrations for different source views with identical SOURCE_OWNER and SOURCE_TABLE column values. This value is set to 0 for physical tables that are defined as sources and is greater than 0 for views that are defined as sources.														
APPLY_QUAL	The Apply qualifier that identifies which Apply program is processing this member.														
SET_NAME	The name of the subscription set that this subscription-set member belongs to.														
CNTL_SERVER	The name of the server where the Apply control tables reside for this Apply program, which is identified by the APPLY_QUAL.														
TARGET_STRUCTURE	A value that identifies the type of target table or view: <table> <tr><td>1</td><td>Source table</td></tr> <tr><td>3</td><td>CCD table</td></tr> <tr><td>4</td><td>Point-in-time table</td></tr> <tr><td>5</td><td>Base aggregate table</td></tr> <tr><td>6</td><td>Change aggregate table</td></tr> <tr><td>7</td><td>Replica table</td></tr> <tr><td>8</td><td>User copy table</td></tr> </table>	1	Source table	3	CCD table	4	Point-in-time table	5	Base aggregate table	6	Change aggregate table	7	Replica table	8	User copy table
1	Source table														
3	CCD table														
4	Point-in-time table														
5	Base aggregate table														
6	Change aggregate table														
7	Replica table														
8	User copy table														
CNTL_ALIAS	The DB2 Universal Database alias corresponding to the Apply control server named in the CNTL_SERVER column.														
PHYS_CHANGE_OWNER	The value in the PHYS_CHANGE_OWNER column from the register (IBMSNAP_REGISTER) table that is associated with the source of this particular subscription-set member.														
PHYS_CHANGE_TABLE	The value in the PHYS_CHANGE_TABLE column from the register (IBMSNAP_REGISTER) table that is associated with the source of this particular subscription-set member.														

# Pruning control table

Table 66. Columns in the pruning control table (continued)

Column name	Description
MAP_ID	A uniqueness factor that provides a shorter, more easily used index into this table. It is also used to associate CAPSTART inserts into the signal table with the appropriate row in the pruning control table.

## schema.IBMSNAP\_PRUNE\_LOCK

Server: Capture control server

Default schema: ASN

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The prune lock table is used to serialize the access of CD tables during a cold start or retention-limit pruning. This table ensures that the Apply program does not access the CD table during these critical phases. There are no rows in this table.

## schema.IBMSNAP\_PRUNE\_SET

Server: Capture control server

Default schema: ASN

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The prune set table tracks the progress of the Capture and Apply programs for each subscription set to help coordinate the pruning of the CD and UOW tables. Unlike the pruning control (IBMSNAP\_PRUNCNTL) table, which has one row for each source-to-target mapping, the prune set table has one row for each subscription set.

Table 67 provides a brief description of the columns in the prune set table.

Table 67. Columns in the prune set table

Column name	Description
TARGET_SERVER	The server name where target tables or views for this set reside.
APPLY_QUAL	The Apply qualifier that identifies which Apply program is processing this set.
SET_NAME	The name of the subscription set.
SYNCHTIME	The Apply program uses this column to record its progress, indicating that it has processed data up to this timestamp for the subscription set.

Table 67. Columns in the prune set table (continued)

Column name	Description
SYNCHPOINT	The Apply program uses this column to record its progress, indicating that it has processed data up to this synchpoint value for the subscription set.

**schema.IBMSNAP\_REG\_EXT (OS/400)****Server:** Capture control server**Default schema:** ASN

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The register extension table is an OS/400-specific table that provides supplemental information for the register (IBMSNAP\_REGISTER) table. For every row in the register table, there is a matching row in the register extension table containing a few additional OS/400-specific columns.

This table is maintained by a trigger program (program QZSNJLV8 in library QDP4) on the register table. The trigger is defined at the time the register table is created.

The information from this table is used to track where and how you defined your replication sources on an OS/400 server.

Table 68 provides a brief description of the columns in the register extension table.

Table 68. Columns in the register extension table

Column name	Description
VERSION	The version of DB2 DataPropagator for iSeries that you used to register the source.
SOURCE_OWNER	The high-level qualifier of the source table or view that you registered.
SOURCE_TABLE	The name of the source table or view that you registered.
SOURCE_NAME	A ten-character system name of the source table or view that you used to issue the commands.
SOURCE_MBR	The name of the source table member, which is used for issuing Receive Journal Entry (RCVJRNE) commands and ALIAS support.
SOURCE_TABLE_RDB	When using remote journals, this column contains the database name of the system where the source table actually resides. For local journals, this column is null.

Table 68. Columns in the register extension table (continued)

Column name	Description								
JRN_LIB	The library name of the journal that the source table uses.								
JRN_NAME	The name of the journal that is used by a source table. An asterisk followed by nine blanks in this column means that the source table is currently not in a journal, and it is not possible for the Capture program to capture data for this source.								
FR_START_TIME	The time when the Apply program began to perform a full refresh.								
SOURCE_VIEW_QUAL	Supports the view of subscriptions by matching the similar column in the register table. This value is set to equal 0 for physical tables that are defined as a source and is greater than 0 for views that are defined as sources. You must have this column to support multiple subscriptions for different source views containing identical SOURCE_OWNER and SOURCE_TABLE column values.								
CMT_BEHAVIOR_CASE	<p>An integer that represents how the application programs that are updating the source table use commitment control. The Capture program uses this value to manage its memory usage for CD rows that it has constructed but is not yet ready to write to the CD tables.</p> <table><tr><td>-1</td><td>The applications' commitment control pattern is not yet established. This is the initial value in the column.</td></tr><tr><td>0</td><td>None of the applications that update the source uses commitment control.</td></tr><tr><td>1</td><td>All of the applications that update the source use commitment control. Therefore, two different applications never update the same source table under commitment control at the same time.</td></tr><tr><td>2</td><td>For concurrent applications that update the source, some use commitment control and others do not. It is possible that there are two applications updating the source table using commitment control concurrently.</td></tr></table>	-1	The applications' commitment control pattern is not yet established. This is the initial value in the column.	0	None of the applications that update the source uses commitment control.	1	All of the applications that update the source use commitment control. Therefore, two different applications never update the same source table under commitment control at the same time.	2	For concurrent applications that update the source, some use commitment control and others do not. It is possible that there are two applications updating the source table using commitment control concurrently.
-1	The applications' commitment control pattern is not yet established. This is the initial value in the column.								
0	None of the applications that update the source uses commitment control.								
1	All of the applications that update the source use commitment control. Therefore, two different applications never update the same source table under commitment control at the same time.								
2	For concurrent applications that update the source, some use commitment control and others do not. It is possible that there are two applications updating the source table using commitment control concurrently.								
MAX_ROWS_BTWN_CMTS	The maximum number of rows that the Capture program can process before it commits data to the CD table.								

schema.IBMSNAP\_REGISTER

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

**Server:** Capture control server

**Default schema:** ASN

The register table contains information about replication sources, such as the names of the replication source tables, their attributes, and the names of the



CD and CCD tables associated with them. A row is automatically inserted into this table every time you define a new replication source table or view for the Capture program to process.

The register table is the place you should look if you need to know how you defined your replication sources.

Table 69 provides a brief description of the columns in the register table.

*Table 69. Columns in the register table*

Column name	Description
SOURCE_OWNER	The high-level qualifier of the source table or view that you registered.
SOURCE_TABLE	The name of the source table or view that you registered.
SOURCE_VIEW_QUAL	This column is used to support multiple registrations for different source views with identical SOURCE_OWNER and SOURCE_TABLE column values. This value is set to 0 for physical tables that are defined as sources, and is greater than 0 for views that are defined as sources.
GLOBAL_RECORD	<p>A flag that indicates whether this row is the global record. In the global record, the SYNCHPOINT and SYNCHTIME columns are set by the Capture program to reflect its progress. If you have not yet run the Capture program, or if you will not be running the Capture program (because you defined the source for full-refresh only replication or because it is a non-DB2 relational source), then there is no global record.</p> <p><b>Y</b>      This row is the global record.</p> <p><b>N</b>      This row is not the global record.</p>
SOURCE_STRUCTURE	<p>A value that identifies the structure of the source table or view:</p> <p><b>1</b>      User table</p> <p><b>3</b>      CCD table</p> <p><b>4</b>      Point-in-time table</p> <p><b>5</b>      Base aggregate table</p> <p><b>6</b>      Change aggregate table</p> <p><b>7</b>      Replica table</p> <p><b>8</b>      User copy table</p>
SOURCE_CONDENSED	<p>A flag that indicates whether the source table is a condensed table, meaning that all rows with the same key are condensed to one row:</p> <p><b>Y</b>      The source is condensed.</p> <p><b>N</b>      The source is not condensed.</p> <p><b>A</b>      The source is a base-aggregate or change-aggregate table.</p>

Table 69. Columns in the register table (continued)

Column name	Description
SOURCE_COMPLETE	<p>A flag that indicates how the source table stores rows of primary key values:</p> <p><b>Y</b>      The source table contains a row for every primary key value of interest.</p> <p><b>N</b>      The source table contains a subset of rows of primary key values.</p>
CD_OWNER	<p>The high-level qualifier of the source's CD table.</p> <p><b>For tables as sources</b> For all registered source tables that are not external CCD tables, this column contains the high-level qualifier of the CD table associated with that source table.</p> <p><b>For views as sources</b> This column contains the high-level qualifier of the CD view.</p> <p><b>For external CCD tables as sources</b> This column is null.</p>
CD_TABLE	<p>The name of the source's CD table.</p> <p><b>For tables as sources</b> For all registered source tables that are not external CCD tables, this column contains the name of the CD table that holds captured updates of the source table.</p> <p><b>For views as sources</b> This column contains the name of the CD view.</p> <p><b>For external CCD tables as sources</b> This column is null.</p>
PHYS_CHANGE_OWNER	<p>The high-level qualifier of the table or view that the Apply program uses for change-capture replication:</p> <p><b>For tables as sources</b> For all registered source tables that are not external CCD tables, this column contains the high-level qualifier of the physical CD table that is associated with that source table.</p> <p><b>For views as sources</b> This column contains the high-level qualifier of the physical CD table that is associated with that source view.</p> <p><b>For external CCD tables as sources</b> This column contains the high-level qualifier of the external CCD table.</p>

Table 69. Columns in the register table (continued)

Column name	Description
PHYS_CHANGE_TABLE	<p>The name of the table or view that the Apply program uses for change-capture replication:</p> <p><b>For tables as sources</b> For all registered source tables that are not external CCD tables, this column contains the name of the physical CD table that is associated with that source table.</p> <p><b>For views as sources</b> This column contains the name of the physical CD table that is associated with that source view.</p> <p><b>For external CCDs as sources</b> This column contains the name of the external CCD table.</p>
CD_OLD_SYNCHPOINT	This column is used for the initial handshake between the Apply program and the Capture program. The Capture program then begins capturing data from this log sequence number in the source log. This column is also used to show that retention-limit pruning has occurred for a CD table. If this value is null, then the registration is inactive.
CD_NEW_SYNCHPOINT	The Capture program advances this column as it inserts new rows into the CD table. The Apply program uses this column to see if there are new changes to be replicated.
DISABLE_REFRESH	<p>A flag that indicates whether full refreshes are allowed:</p> <p><b>0</b> Full refreshes are allowed.</p> <p><b>1</b> Full refreshes are prevented.</p>
CCD_OWNER	For a source that has an internal CCD table associated with it, this column contains the high-level qualifier of the internal CCD. For an external CCD table, this column is null.
CCD_TABLE	For a source that has an internal CCD table associated with it, this column contains the name of the internal CCD. For an external CCD table, this column is null.
CCD_OLD_SYNCHPOINT	The log sequence number when the CCD table was reinitialized. This column is related to full-refresh processing against CCD tables. The value in this column needs to be changed only when the CCD table is initially or subsequently fully refreshed. This value can be much older than any row remaining in the CCD table. If this column is not maintained, the Apply program using the CCD table as a replication source will not know that the CCD table was reinitialized, so it will fail to reinitialize complete copies of the CCD source.

Table 69. Columns in the register table (continued)

Column name	Description
SYNCHPOINT	In the global row (where GLOBAL_RECORD = Y), the synchpoint represents the log sequence number of the last log or journal record processed by the Capture program. In any row in the IBMSNAP_REGISTER table that contains registration information about a CCD table (internal or external), the synchpoint value is advanced by the program that maintains the CCD table to indicate that there is new data available in that CCD table.
SYNCHTIME	In the global row (where GLOBAL_RECORD = Y), the synchtime represents the timestamp from the last log or journal record processed by the Capture program. If the Capture program has reached the end of the DB2 log, the synchtime is advanced to the current DB2 timestamp. In any row in the IBMSNAP_REGISTER table that contains registration information about a CCD table (internal or external), the synchtime value is advanced by the program that maintains the CCD table to indicate the currency of data available in that CCD table.
CCD_CONDENSED	<p>A flag that indicates whether the internal CCD that is associated with this source is condensed, meaning that all rows with the same key are condensed to one row:</p> <p><b>Y</b>        The internal CCD is condensed.</p> <p><b>N</b>        The internal CCD is not condensed.</p> <p><b>NULL</b>    No internal CCD table is defined for this source.</p>
CCD_COMPLETE	<p>A flag that indicates whether the internal CCD table that is associated with this source is complete, meaning that it initially contained all the rows from the source table:</p> <p><b>N</b>        The internal CCD is not complete.</p> <p><b>NULL</b>    No internal CCD table is defined for this source.</p>
ARCH_LEVEL	The architectural level of the definition in the row. This level is defined by IBM, and for Version 8 is 0801.
DESCRIPTION	A description of the replication source.
BEFORE_IMG_PREFIX	<p>The one-character prefix that identifies before-image column names in the CD table. The combination of the before-image prefix and the CD column name must be unambiguous, meaning that a prefixed CD column name cannot be the same as a current or potential after-image column name. The length in bytes of the BEFORE_IMG_PREFIX is:</p> <p><b>1</b>        For an ASCII or an EBCDIC single byte prefix character.</p> <p><b>2</b>        For an ASCII double byte prefix character.</p> <p><b>4</b>        For an EBCDIC DBCS prefix character. This length allows for shift-in and shift-out characters.</p>

Table 69. Columns in the register table (continued)

Column name	Description
CONFLICT_LEVEL	<p>A flag that indicates the level of conflict detection for this source:</p> <p><b>0</b>      The Apply program does not check for conflicts. Data consistency must be enforced by your application to avoid potential conflicting updates.</p> <p><b>1</b>      Standard detection with cascading transaction rejection. The Apply program checks for conflicts based on the changes captured to this point. The Apply program will reverse any conflicting transaction at the replica, as well as any transactions with dependencies on the conflicting transaction. Changes captured after the Apply program begins conflict detection will not be checked during this Apply cycle.</p> <p><b>2</b>      Enhanced detection with cascading transaction rejection. The Apply program waits until the Capture program captures all changes from the log or journal (see description of the SYNCHTIME column) and then does a standard conflict detection as when set to 1. During the wait time, the Apply program holds a LOCK on the source tables to ensure that no changes are made during the conflict detection process.</p>
CHG_UPD_TO_DEL_INS	<p>A flag that indicates how the Capture program stores updates in the CD table.</p> <p><b>Y</b>      The Capture program stores updates using two rows in the CD table, one for the delete and one for the insert. The Apply program processes the delete first and the insert second. When this Y flag is set, every update to a replication source is stored in the CD table using two rows. This flag ensures that updates made to partitioning columns or columns referenced by a subscription-set predicate are processed correctly.</p> <p><b>N</b>      Each update to the source table is stored in a single row in the CD table.</p>
CHGONLY	<p>A flag that indicates whether the Capture program captures all changes that occur at the source or only changes that occur in registered columns. Typically you should have this option set to Y to minimize the number of rows that the Capture program inserts into the CD table, but you may want to set this option to N in order to track exactly which rows in the source table were updated. For example, you may just be capturing the primary key column values to audit which rows have been changed in a source table.</p> <p><b>Y</b>      The Capture program only captures changes that occur in registered columns in the source table.</p> <p><b>N</b>      The Capture program captures changes from all columns in the source table.</p>

Table 69. Columns in the register table (continued)

Column name	Description
RECAPTURE	<p>This column is for update-anywhere replication and contains a flag that indicates whether changes that originate from a table or view are recaptured and forwarded to other tables or views.</p> <p>For tables at the master site:</p> <p><b>N</b> Updates to the master that were applied from a replica are not recaptured and will not be replicated to other replicas.</p> <p><b>Y</b> Updates to the master that were applied from a replica and will be replicated to other replicas.</p> <p>For tables at a replica site:</p> <p><b>Y</b> Updates to the replica that were applied from the master are recaptured and are available to be replicated to another table that uses the replica as its source.</p> <p><b>N</b> Updates to the replica that were applied from the master are not recaptured.</p>
OPTION_FLAGS	Reserved for future options of DB2 replication. Currently this column contains the default value of NNNN.
STOP_ON_ERROR	<p>A flag that indicates whether the Capture program will terminate or just stop processing the registration if it encounters errors while trying to start, initiate, reinitiate, or insert a row into the CD table:</p> <p><b>Y</b> The Capture program terminates when an error occurs while it is trying to start, initiate, reinitiate, or insert a row into the CD table.</p> <p><b>N</b> The Capture program stops the registration but does not terminate when an error occurs while it is trying to start, initiate, reinitiate, or insert a row into the CD table; it continues to process other registrations.</p>
STATE	<p>A flag that indicates what state the registration is in:</p> <p><b>S</b> The Capture program has stopped processing this registration. The Apply program will not work with this registration until you repair the registration and place it in the I (inactive) state.</p> <p><b>A</b> The registration is active.</p> <p><b>I</b> The registration is inactive.</p>
STATE_INFO	If the Capture program stopped processing the registration, this column contains the error message that was issued regarding the failure.

**schema.IBMSNAP\_REG\_SYNC (non-DB2 relational)****Server:** Capture control server**Default schema:** ASN

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The register synchronization table uses an update trigger to initiate an update of the SYNCHPOINT value for all the rows in the register (IBMSNAP\_REGISTER) table when the Apply program is preparing to fetch data from a non-DB2 relational data source.

Table 70 provides a brief description of the columns in the register synchronization table.

*Table 70. Register synchronization table columns*

Column name	Description
TRIGGER_ME	A flag of Y that indicates whether a trigger was initiated to update the SYNCHPOINT value for all rows in the register table.
TIMESTAMP	For Microsoft SQL Server and Sybase sources, this column contains the unique number that is generated by the system when an update occurs on a timestamp column at that table. This value is used to derive the SYNCHPOINT value that is recorded in the register (IBMSNAP_REGISTER) table.

**schema.IBMSNAP\_RESTART****Server:** Capture control server**Default schema:** ASN

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data. If you delete the row from this table, the Capture program is forced to cold start.

The restart table contains information that enables the Capture program to restart from the earliest required log or journal record. This table replaces the warm start table from DB2 replication Version 7 and earlier versions. It contains one row, which is updated at every commit point; therefore, the Capture program can always restart from exactly the right place without recapturing information that it already processed and inserted into the CD and UOW tables.

If you have never started the Capture program, then this table is empty and the Capture program must perform a cold start.

The following two tables show operating system-specific layouts of the restart table:

Table 71. Columns in the restart table for UNIX, Windows, and z/OS

Column name	Description
MAX_COMMITSEQ	The maximum log sequence number value (IBMSNAP_COMMITSEQ) that the Capture program has committed to the CD and UOW tables.
MAX_COMMIT_TIME	The timestamp that is associated with the log sequence number in the MAX_COMMITSEQ column.
MIN_INFLIGHTSEQ	The log sequence number at which the Capture program starts during a warm restart. This value represents the earliest log sequence number that the Capture program found for which a commit or abort record has not yet been found.
CURR_COMMIT_TIME	The local current timestamp when this table was updated by the Capture program.
CAPTURE_FIRST_SEQ	The log sequence number that is associated with the recovery log that the Capture program started from during the last cold start it performed. This value is used to detect and alert the user if a database RESTORE occurred, which may require the Capture program to perform a cold start since the database log manager may reuse the log sequence numbers during certain RESTORE operations.

For OS/400, the restart table is used to determine the starting time of the **RCVJRNE** (Receive Journal Entry) command. A row is inserted into the restart table for each journal that is used by a replication source or a group of replication sources.

Table 72. Columns in the restart table for OS/400

Column name	Description
MAX_COMMITSEQ	The journal record number of the most current commit from the UOW table.
MAX_COMMIT_TIME	The timestamp that is associated with the journal record number in the MAX_COMMITSEQ column, or the current timestamp if the Capture program is caught up with the logs and has no work to perform.
MIN_INFLIGHTSEQ	The log sequence number that the Capture program starts from during a warm restart.
CURR_COMMIT_TIME	The current timestamp at the point when this table is updated.
CAPTURE_FIRST_SEQ	The journal record number that the Capture program starts from after a cold start.
UID	A unique number that is used as a prefix for the contents of the IBMSNAP_UOWID column located in the UOW table.



Table 72. Columns in the restart table for OS/400 (continued)

Column name	Description
SEQNBR	The sequence number of the last journal entry that the Capture program processed.
JRN_LIB	The library name of the journal that the Capture program is processing.
JRN_NAME	The name of the journal that the Capture program is processing.
STATUS	A flag that indicates whether the Capture program is processing a particular journal job:  <div> <div>Y</div> <div>The Capture program is processing the journal job.</div> </div> <div> <div>N</div> <div>The Capture program is not processing the journal job.</div> </div>

**schema.IBMSNAP\_SEQTABLE (Informix)****Server:** Capture control server**Default schema:** ASN

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The sequencing table contains a sequence of unique numbers that DB2 replication uses as the equivalent of log sequence numbers for Informix tables. These unique identifiers are used in the register (IBMSNAP\_REGISTER) table in place of synchpoint values so that the Capture program, Apply program, and Replication Alert Monitor can communicate how far along they have gotten during their last cycle.

Table 73 provides a brief description of the column in the sequencing table.

Table 73. Column in the sequencing table

Column name	Description
SEQ	A unique number used as the log or journal identifiers (synchpoints) for Informix tables.

**schema.IBMSNAP\_SIGNAL****Server:** Capture control server**Default schema:** ASN

This table contains information that you can update by using SQL.

The signal table stores signals that prompt the Capture program to perform certain actions. The signals are entered by either you or the Apply program.

The signal table is created with the DATA CAPTURE CHANGES attribute, which means that all insert, update, and delete operations performed on this table are visible to the Capture program as log records read from the DB2 recovery log. The Capture program ignores all update and delete log records for the signal table, but it recognizes all validly created and committed log records of signal inserts as "signals" that require its attention. The actions that the Capture program performs for a log record from a signal insert depends on what is specified in the signal table for that insert. The values in the signal table provide the instructions to the Capture program regarding the desired action.

Records in this table with a SIGNAL\_STATE value of C for complete or records with a timestamp eligible for retention-limit pruning are deleted when the Capture program prunes.

Table 74 provides a brief description of the columns in the signal table.

Table 74. Columns in the signal table

Column name	Description
SIGNAL_TIME	A timestamp that is used to uniquely identify the row. The Capture program uses this unique value to find the correct row in the signal table to indicate when it has completed processing the Capture signal. This timestamp column is created as NOT NULL WITH DEFAULT, and therefore a Capture signal can generally be inserted in such a way that DB2 supplies the current timestamp as the SIGNAL_TIME value.
SIGNAL_TYPE	A flag that indicates the type of signal that was posted:  <b>CMD</b> A signal posted by you, the Apply program, or another application, which is a well known system command or signal. See the SIGNAL_SUBTYPE column for this table for a list of the available signal subtypes.  <b>USER</b> A signal posted by you or another user. The Capture program updates the value in the SIGNAL_LSN column with the LSN from the log of when the signal was inserted, and it updates the value in the SIGNAL_STATE column to from P (pending) to R (received).

Table 74. Columns in the signal table (continued)

Column name	Description
SIGNAL_SUBTYPE	<p>The action that the Capture program performs when a signal from a system command (SIGNAL_TYPE = CMD) occurs.</p> <p><b>CAPSTART</b></p> <p>The Capture program starts capturing changes at the registered source for a particular subscription-set member, which is identified by the Apply qualifier in the SIGNAL_INPUT_IN column. For example, the Apply program issues this signal before it performs a full refresh on all target tables in the set to let the Capture program know that the set is ready to begin change-capture replication. The Apply program posts this signal.</p> <p><b>STOP</b></p> <p>The Capture program stops capturing changes and terminates. This command can only be issued by you, not the Apply program.</p> <p><b>CAPSTOP</b></p> <p>The Capture program stops capturing changes for a particular registered source, which is identified by <i>source_owner.source_table</i> in the SIGNAL_INPUT_IN column. This command can only be issued by you, not the Apply program.</p> <p><b>UPDANY</b></p> <p>The Apply program (identified by the Apply qualifier in the SIGNAL_INPUT_IN column) lets the Capture program know that it is working with two Capture programs in an update-anywhere configuration. The Apply program posts this signal.</p> <p>When the signal type is USER, the signal subtype is not used or recognized by the Capture program and therefore is not a required field. It can be set to any value that you want.</p>

Table 74. Columns in the signal table (continued)

Column name	Description
SIGNAL_INPUT_IN	<p>If the SIGNAL_TYPE = USER, then this column contains user-defined input. If the SIGNAL_TYPE = CMD, then the meaning of this value depends on the SIGNAL_SUBTYPE for this signal:</p> <p><b>CMD + CAPSTART</b> The mapping identifier. Because the Capture triggers and not the Capture program process non-DB2 relational sources, there is an after-insert SIGNAL_TRIGGER that updates the prune control (IBMSNAP_PRUNCNTL) table with the next value in the sequence.</p> <p><b>CMD + UPDANY</b> The Apply qualifier that identifies the Apply program in the update-anywhere configuration.</p> <p><b>CMD + CAPSTOP</b> The name of the source owner and source table that the Capture program should stop capturing changes for. (source_owner.source_table)</p>
SIGNAL_STATE	<p>A flag that indicates the status of the signal:</p> <p><b>P</b> The signal is pending; the Capture program has not received it yet. When you post a signal, set the SIGNAL_STATE to P.</p> <p><b>R</b> The Capture program has received the signal. The Capture program sets the SIGNAL_STATE set to R (instead of changing it to C for complete) when it receives a signal where SIGNAL_TYPE = USER, or one where SIGNAL_TYPE = CMD and SIGNAL_SUBTYPE = STOP.</p> <p><b>C</b> The Capture program has completed processing the signal. The Capture program sets this value to C when SIGNAL_TYPE = CMD for all SIGNAL_SUBTYPE values except STOP.</p>
SIGNAL_LSN	The log sequence number of the commit record. This value is set only by the Capture program.

schema.IBMSNAP\_UOW

**Server:** Capture control server

**Default schema:** ASN

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The unit-of-work (UOW) table provides additional information about transactions that have been committed to a source table. For all target table types other than user copy, the Apply program joins the UOW and change

data (CD) tables based on matching IBMSNAP\_COMMITSEQ values when it applies changes to the target tables. If you cold start the Capture program, all of this table's entries are deleted.

**For OS/400:** Because Capture for iSeries can start capturing data for a subset of the replication sources, it does not delete all the rows in the UOW table if you do a partial cold start.

The Capture program requires that there is one UOW table for each Capture schema. The Capture program inserts one new row into this table for every log or journal record that is committed at the replication source.

**For OS/400:** There are some user programs that do not use commitment control. In such cases, Capture for iSeries arbitrarily inserts a new UOW row after a number of rows are written to the CD table. This artificial commitment boundary helps reduce the size of the UOW table.

The Capture program also prunes the UOW table based on information that the Apply program inserts into the prune set (IBMSNAP\_PRUNE\_SET) table.

**For OS/400:** The UOW table is pruned by retention limits, not information from the prune set (IBMSNAP\_PRUNE\_SET) table.

Table 75 provides a brief description of the columns in the UOW table.

Table 75. Columns in the UOW table

Column name	Description
IBMSNAP_UOWID	The unit-of-work identifier from the log record header for this unit of work. You can select that this column be part of a noncomplete CCD target table.
IBMSNAP_COMMITSEQ	The log record sequence number of the captured commit statement. For all target table types other than user copy, the Apply program joins the UOW and CD tables based on the values in this column when it applies changes to the target tables.
IBMSNAP_LOGMARKER	The time (at the Capture control server) that the data was committed.
IBMSNAP_AUTHTKN	The authorization token that is associated with the transaction. This ID is useful for database auditing. For DB2 Universal Database for z/OS, this column is the correlation ID. For DB2 Universal Database for iSeries, this column is the job name of the job that caused a transaction. This column is not automatically copied to other tables; you must select it and copy it as a user data column. You can select that this column be part of a noncomplete CCD target table.

Table 75. Columns in the UOW table (continued)

Column name	Description
IBMSNAP_AUTHID	The authorization ID that is associated with the transaction. It is useful for database auditing. The length is 18 characters. If you supply a longer value, it is truncated. For DB2 Universal Database for z/OS, this column is the primary authorization ID. For DB2 Universal Database for iSeries, this column has the name of the user profile ID under which the application that caused the transaction ran. This column holds the ten-character ID padded with blanks. This column is not automatically copied to other tables; you must select it and copy it as a user data column. You can select for this column to be part of a noncomplete CCD target table.
IBMSNAP_REJ_CODE	<p>A flag that indicates whether any rows were rejected and rolled back. This value is set only during update-anywhere replication if conflict detection is specified as standard or enhanced when you defined your replication source. You can select that this column be part of a noncomplete CCD target table.</p> <p><b>0</b> No known conflicts occurred in the transaction.</p> <p><b>1</b> A conflict occurred because the same row in the master and replica was updated. The transaction at the replica was rejected and rolled back.</p> <p><b>2</b> The transaction was rejected and rolled back because it was dependent on a prior transaction that was rejected. The prior transaction was rejected because the same row in the master and replica was updated, and the transaction at the replica was rejected and rolled back.</p> <p><b>3</b> The transaction was rejected and rolled back because it contained at least one referential-integrity constraint violation. Because this transaction violates the referential constraints defined on the source table, the Apply program will mark this subscription set as failed. Updates cannot be copied until you correct the referential integrity definitions.</p> <p><b>4</b> The transaction was rejected and rolled back because it was dependent on a prior transaction that was rejected. The prior transaction was rejected because it contained at least one referential-integrity constraint violation.</p>
IBMSNAP_APPLY_QUAL	The Apply qualifier that identifies which Apply program applied the changes. You can select that this column be part of a noncomplete CCD target table.

## Tables at the Apply control server and their column descriptions

This section provides greater detail of each table stored at the Apply control server. It also lists and briefly describes the columns in each table. The control tables are listed in alphabetical order, and the columns are listed in the order that they appear in each table from left to right.

### ASN.IBMSNAP\_APPENQ

**Server:** Apply control server

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The Apply enqueue table is used to ensure that only one Apply program is running per Apply qualifier. The Apply program exclusively locks a row in this table until the Apply program is shut down. This table is not used in OS/400.

Table 76 provides a brief description of the column in the Apply enqueue table.

*Table 76. Column in the Apply enqueue table*

Column name	Description
APPLY_QUAL	Uniquely identifies a group of subscription sets that are processed by the same Apply program. This value is case sensitive. You must specify this value when you define a subscription set.

### ASN.IBMSNAP\_APPLY\_JOB (OS/400)

**Server:** Apply control server

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The Apply job table, which is OS/400-specific, is used to guarantee a unique APPLY\_QUAL value for all instances of the Apply program running at the Apply control server. A row is added to this table every time an instance of the Apply program is started. If you start a new instance of the Apply program with an APPLY\_QUAL value that already exists, the start command fails.

Table 77 on page 514 provides a brief description of the columns in the Apply job table.

## ASN.IBMSNAP\_APPLY\_JOB (OS/400)

Table 77. Columns in the Apply job table

Column name	Description
APPLY_QUAL	A unique identifier for a group of subscription sets. This value is supplied by the user when defining a subscription set. Each instance of the Apply program is started with an APPLY_QUAL value. This value is used during update-anywhere replication to prevent circular replication of the changes made by the Apply program.
CONTROL_SERVER	The name of the database where the Apply control tables and view are defined.
JOB_NAME	The fully qualified name of the job that wrote this trace entry:  <b>Position 1-10</b> APPLY_QUAL  <b>Position 11-20</b> The ID of the user who started the Apply program  <b>Position 21-26</b> The job number
USER_NAME	The name of the user who started a new instance of the Apply program.
JOB_NUMBER	The job number of the current job for a particular journal. If the journal is not active, this column contains the job number of the last job that was processed.

## ASN.IBMSNAP\_APPLYTRACE

**Server:** Apply control server

The Apply trace table contains important messages from the Apply program. The Apply program does not automatically prune this table, but you can easily automate pruning by adding an SQL statement that runs after one of the subscription sets.

Table 78 provides a brief description of the column in the Apply trace table.

Table 78. Columns in the Apply trace table

Column name	Description
APPLY_QUAL	Uniquely identifies which Apply program inserted the message.
TRACE_TIME	The time at the Apply control server when the row was inserted into this table.
OPERATION	The type of Apply program operation, for example, initialization, apply, or error condition.
DESCRIPTION	The message ID followed by the message text. The message ID is the first seven characters of the DESCRIPTION column. The message text starts at the ninth position of the DESCRIPTION column.



**ASN.IBMSNAP\_APPLYTRAIL**

**Server:** Apply control server

The Apply trail table contains audit trail information of all subscription set cycles performed by the Apply program. This table records a history of updates that are performed against subscriptions. It is a repository of diagnostic and performance statistics. The Apply trail table is one of the best places to look if a problem occurs with the Apply program. The Apply program does not automatically prune this table, but you can easily automate pruning by adding an after SQL statement to one of the subscription sets.

Table 79 provides a brief description of the columns in the Apply trail table.

*Table 79. Columns in the Apply trail table*

Column name	Description
APPLY_QUAL	Uniquely identifies which Apply program was processing the subscription set.
SET_NAME	The name of the subscription set that the Apply program was processing.
SET_TYPE	The value that appeared in the SET_TYPE column of the subscription set (IBMSNAP_SUBS_SET) table after the most recent Apply cycle. See “ASN.IBMSNAP_SUBS_SET” on page 526 for a description of what each value means.
WHOS_ON_FIRST	<p>The following values are used to control the order of processing in update-anywhere replication scenarios.</p> <p><b>F</b> (first) The source table is the replica and the target table is the master. In the case of update conflicts between the replica and the master table, the replica will have its conflicting transactions rejected. F is not used for read-only subscriptions; it is used for update anywhere.</p> <p><b>S</b> (second) The source table is the master table or other source, and the target table is the replica or other copy. In the case of update conflicts between the master and the replica table, the replica will have its conflicting transactions rejected. S is used for all read-only subscriptions.</p>

## ASN.IBMSNAP\_APPLYTRAIL

Table 79. Columns in the Apply trail table (continued)

Column name	Description
ASNLOAD	<p>The value used to start the Apply program:</p> <p><b>Y</b> Indicates that the Apply program was started with the parameter <b>loadxit=y</b> causing the ASNLOAD user exit routine to be called to perform a full refresh on a subscription set.</p> <p><b>N</b> Indicates that the ASNLOAD exit routine was not called because either a full refresh was not needed or the Apply program was not started with the <b>loadxit</b> parameter.</p> <p><b>NULL</b> Indicates that an Apply program error occurred before the Apply program could determine whether the ASNLOAD exit routine should be called.</p>
FULL_REFRESH	<p>A flag that indicates whether a full refresh occurred:</p> <p><b>Y</b> Indicates that a full refresh was done for a subscription set.</p> <p><b>N</b> Indicates that a full refresh was not done for a subscription set.</p> <p><b>NULL</b> Indicates that an error occurred before the Apply program could determine whether or not a full refresh was needed.</p>
EFFECTIVE_MEMBERS	<p>The number of subscription-set members that are changed during an Apply cycle, either by a full refresh or by the replication of inserts, updates, and deletes. This number ranges between zero and the number of defined subscription-set members.</p>
SET_INSERTED	<p>The total number of rows inserted into subscription-set members during the subscription cycle.</p>
SET_DELETED	<p>The total number of rows deleted from subscription-set members during the subscription cycle.</p>
SET_UPDATED	<p>The total number of rows updated in subscription-set members during the subscription cycle.</p>
SET_REWORKED	<p>The total number of rows that the Apply program reworked during the last cycle. The Apply program reworks changes under the following conditions:</p> <ul style="list-style-type: none"><li>• If an insert fails because the row already exists in the target table, the Apply program converts the insert to an update of the existing row.</li><li>• If the update fails because the row does not exist in the target table, the Apply program converts the update to an insert.</li></ul>
SET_REJECTED_TRXS	<p>The total number of transactions that were rejected due to an update-anywhere conflict. This column is used only for update-anywhere subscription sets where conflict detection is defined as standard or advanced.</p>

Table 79. Columns in the Apply trail table (continued)

Column name	Description
STATUS	<p>A value that represents the work status for the Apply program after a given cycle:</p> <ul style="list-style-type: none"> <li><b>-1</b> The replication failed. The Apply program backed out the entire set of rows that it had applied, and no data was committed. If the startup parameter <code>SQLERRCONTINUE = Y</code>, the <code>SQLSTATE</code> that is returned to the Apply program during the last cycle is <i>not</i> one of the acceptable errors you indicated in the input file for <code>SQLERRCONTINUE</code> (<i>apply_qualifier.SQS</i>).</li> <li><b>0</b> The Apply program processed the subscription set successfully. If the startup parameter <code>SQLERRCONTINUE = Y</code>, the Apply program did not encounter any SQL errors that you indicated for the <code>SQLERRCONTINUE</code> startup parameter (in <i>apply_qualifier.SQS</i>) and did not reject any rows.</li> <li><b>2</b> The Apply program is processing the subscription set in multiple cycles. It successfully processed a single logical subscription that was divided according to the <code>MAX_SYNC_MINUTES</code> control column.</li> <li><b>16</b> The Apply program processed the subscription set successfully and returned a status of 0; however, it encountered some SQL errors that you indicated for the <code>SQLERRCONTINUE</code> startup parameter (in <i>apply_qualifier.SQS</i>) and rejected some of the rows. See the <i>apply_qualifier.ERR</i> file for details about the rows that failed.   <b>Example:</b> You set <code>SQLERRCONTINUE = Y</code> and indicate that the allowable SQL state is 23502 (SQL code -407). A 23502 error occurs, but no other errors occur. The Apply program finishes processing the subscription set, and it sets the status to 16. On the next execution, a 23502 error occurs, but then a 07006 (SQL code -301) occurs. Now the Apply program stops processing the subscription set, backs out the entire set of rows it had applied, and sets the status to -1 (because no data was committed).</li> <li><b>18</b> The Apply program is processing the subscription set in multiple cycles and returned a status of 2, which means that it successfully processed a single logical subscription that was divided according to the <code>MAX_SYNC_MINUTES</code> control column. However, it encountered some SQL errors that you indicated for the <code>SQLERRCONTINUE</code> startup parameter (in <i>apply_qualifier.SQS</i>) and rejected some of the rows. See the <i>apply_qualifier.ERR</i> file for details about the rows that failed.</li> </ul>

Table 79. Columns in the Apply trail table (continued)

Column name	Description
LASTRUN	The estimated time that the last subscription began. The Apply program sets the LASTRUN value each time a subscription set is processed. It is the approximate time at the Apply control server that the Apply program begins processing the subscription set.
LASTSUCCESS	The Apply control server timestamp for the beginning of the last successful processing of a subscription set.
SYNCHPOINT	The Apply program uses this column to record its progress, indicating that it has processed data up to this synchpoint value for the subscription set.
SYNCHTIME	The Apply program uses this column to record its progress, indicating that it has processed data up to this timestamp for the subscription set.
SOURCE_SERVER	The DB2 Universal Database database name where the source tables and views are defined.
SOURCE_ALIAS	The DB2 Universal Database alias corresponding to the source server named in the SOURCE_SERVER column.
SOURCE_OWNER	The high-level qualifier of the source table or view that the Apply program was processing. This value is set only when the Apply cycle fails.
SOURCE_TABLE	The name of the source table or view that the Apply program was processing. This value is set only when the Apply cycle fails.
SOURCE_VIEW_QUAL	The value of the source view qualifier for the source table or view that the Apply program was processing. This value is set only when the Apply cycle fails.
TARGET_SERVER	The database name of the server where target tables or views are stored.
TARGET_ALIAS	The DB2 Universal Database alias corresponding to the target server named in the TARGET_SERVER column.
TARGET_OWNER	The high-level qualifier of the target table that the Apply program was processing. This value is set only when the Apply cycle fails.
TARGET_TABLE	The name of the target table that the Apply program was processing. This value is set only when the Apply cycle fails.
CAPTURE_SCHEMA	The schema name of the Capture server tables for this subscription set.
TGT_CAPTURE_SCHEMA	If the target table is also the source for another subscription set (such as an external CCD table in a multi-tier configuration or a replica table in an update-anywhere configuration), this column contains the Capture schema that will be used when the table is acting as a source.
FEDERATED_SRC_SRVR	The name of the federated remote server that is the source for the subscription set, which applies only to non-DB2 relational sources.
FEDERATED_TGT_SRVR	The name of the federated remote server that is the target for the subscription set, which applies only to non-DB2 relational target servers.

Table 79. Columns in the Apply trail table (continued)

Column name	Description
JRN_LIB	This column, which applies only to OS/400 Capture servers, is the library name of the journal that the source table uses.
JRN_NAME	This column, which applies only to OS/400 Capture servers, is the name of the journal used by a source table. An asterisk followed by nine blanks in this column means that the source table is currently not in a journal, in which case it is not possible to capture data for this source table.
COMMIT_COUNT	The value of the COMMIT_COUNT from the last Apply cycle, which is recorded in the subscription sets (IBMSNAP_SUBS_SET) table.
OPTION_FLAGS	Reserved for future options of DB2 replication. Currently this column contains the default value of NNNN.
EVENT_NAME	A unique character string used to represent the event that triggered the set to be processed.
ENDTIME	The timestamp at the Apply control server when the Apply program finished processing the subscription set. To find out how long a set took to process, subtract LASTRUN from ENDTIME.
SOURCE_CONN_TIME	The timestamp at the Capture control server when the Apply program first connects to fetch source data.
SQLSTATE	The SQL state code for a failed execution. Otherwise, NULL.
SQLCODE	The SQL error code for a failed execution. Otherwise, NULL.
SQLERRP	The database product identifier of the server where an SQL error occurred that caused a failed execution. Otherwise, NULL.
SQLERRM	The SQL error information for a failed execution.
APPERRM	The Apply error message ID and text for a failed execution.

## ASN.IBMSNAP\_SUBS\_COLS

**Server:** Apply control server

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The subscription columns table contains information about the columns of the subscription-set members that are copied in a subscription set. Rows are automatically inserted into or deleted from this table when information changes in one or more columns of a source and target table pair.

Use this table if you need information about specific columns in a subscription-set member.

Table 80 provides a brief description of the columns in the subscription columns table.

Table 80. Columns in the subscription columns table

Column name	Description
APPLY_QUAL	Uniquely identifies which Apply program processes this subscription-set member.
SET_NAME	The name of a subscription set that this member belongs to.
WHOS_ON_FIRST	<p>The following values are used to control the order of processing in update-anywhere replication scenarios.</p> <p><b>F</b> (first) The source table is the replica and the target table is the master. In the case of update conflicts between the replica and the master table, the replica will have its conflicting transactions rejected. F is not used for read-only subscriptions; it is used for update anywhere.</p> <p><b>S</b> (second) The source table is the master table or other source, and the target table is the replica or other copy. In the case of update conflicts between the master and the replica table, the replica will have its conflicting transactions rejected. S is used for all read-only subscriptions.</p>
TARGET_OWNER	The high-level qualifier for a target table or view.
TARGET_TABLE	The table or view to which data is being applied.
COL_TYPE	<p>A flag that indicates the type of column:</p> <p><b>A</b> An after-image column.</p> <p><b>B</b> A before-image column.</p> <p><b>C</b> A computed column or an SQL expression using scalar functions.</p> <p><b>D</b> A DATALINK column.</p> <p><b>F</b> A computed column using column functions.</p> <p><b>L</b> A LOB indicator value.</p> <p><b>P</b> A before-image predicate column.</p> <p><b>R</b> A relative record number column, provided by the system and used as a primary key column. Used only by DB2 DataPropagator for iSeries.</p>
TARGET_NAME	<p>The name of the target table or view column. It does not need to match the source column name.</p> <p>Internal CCD column names cannot be renamed. They must match the table column names.</p>

Table 80. Columns in the subscription columns table (continued)

Column name	Description
IS_KEY	A flag that indicates whether the column is part of the target key, which can be either a unique index or primary key of a condensed target table:  Y        The column is all or part of the target key.  N        The column is not part of the target key.
COLNO	The numeric location of the column in the original source, to be preserved relative to other user columns in displays and subscriptions.
EXPRESSION	The source column name or an SQL expression used to create the target column contents.

**ASN.IBMSNAP\_SUBS\_EVENT****Server:** Apply control server

This table contains information that you can update using SQL.

The subscription events table contains information about the event triggers that are associated with a subscription set. It also contains names and timestamps that are associated with the event names. You insert a row into this table when you create a new event to start an Apply program. See “Event-based scheduling” on page 74.

Table 81 provides a brief description of the columns in the subscription events table.

Table 81. Columns in the subscription events table

Column name	Description
EVENT_NAME	The unique identifier of an event. This identifier is used to trigger replication for a subscription set.
EVENT_TIME	An Apply control server timestamp of a current or future posting time. User applications that signal replication events provide the values in this column.
END_SYNCPOINT	A log sequence number that tells the Apply program to apply only data that has been captured up to this point. You can find the exact END_SYNCPOINT that you want to use by referring to the signal table and finding the precise log sequence number associated with a timestamp. Any transactions that are committed beyond this point in the log are not replicated until a later event is posted. If you supply values for END_SYNCPOINT and END_OF_PERIOD, the Apply program uses the END_SYNCPOINT value because it then does not need to perform any calculations from the control tables to find the maximum log sequence number to replicate.

ASN.IBMSNAP\_SUBS\_EVENT

Table 81. Columns in the subscription events table (continued)

Column name	Description
END_OF_PERIOD	A timestamp used by the Apply program, which applies only data that has been logged up to this point. Any transactions that are committed beyond this point in the log are not replicated until a later event is posted.

ASN.IBMSNAP\_SUBS\_MEMBR

Server: Apply control server

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The subscription members table contains information about the individual source and target table pairs defined for a subscription set. A single row is automatically inserted into this table when you add a subscription set member.

Use this table to identify a specific source and target table pair within a subscription set.

Table 82 provides a brief description of the columns in the subscription member table.

Table 82. Columns in the subscription member table

Column name	Description
APPLY_QUAL	Uniquely identifies which Apply program processes this subscription-set member.
SET_NAME	The name of the subscription set that this member belongs to.
WHOS_ON_FIRST	The following values are used to control the order of processing in update-anywhere replication scenarios.  F (first) The source table is the replica and the target table is the master. In the case of update conflicts between the replica and the master table, the replica will have its conflicting transactions rejected. F is not used for read-only subscriptions; it is used for update anywhere.  S (second) The source table is the master table or other source, and the target table is the replica or other copy. In the case of update conflicts between the master and the replica table, the replica will have its conflicting transactions rejected. S is used for all read-only subscriptions.
SOURCE_OWNER	The high-level qualifier for the source table or view for this member.
SOURCE_TABLE	The name of the source table or view for this member.



Table 82. Columns in the subscription member table (continued)

Column name	Description														
SOURCE_VIEW_QUAL	Supports the view of physical tables by matching the similar column in the register table. This value is set to 0 for physical tables that are defined as sources and is greater than 0 for views that are defined as sources. This column is used to support multiple subscriptions for different source views with identical SOURCE_OWNER and SOURCE_TABLE column values.														
TARGET_OWNER	The high-level qualifier for the target table or view for this member.														
TARGET_TABLE	The name of the target table or view for this member.														
TARGET_CONDENSED	A flag that indicates: <table> <tr> <td>Y</td><td>For any given primary key value, the target table shows only one row.</td></tr> <tr> <td>N</td><td>All changes must remain to retain a complete update history.</td></tr> <tr> <td>A</td><td>The target table is a base aggregate or change aggregate tables.</td></tr> </table>	Y	For any given primary key value, the target table shows only one row.	N	All changes must remain to retain a complete update history.	A	The target table is a base aggregate or change aggregate tables.								
Y	For any given primary key value, the target table shows only one row.														
N	All changes must remain to retain a complete update history.														
A	The target table is a base aggregate or change aggregate tables.														
TARGET_COMPLETE	A flag that indicates: <table> <tr> <td>Y</td><td>The target table contains a row for every primary key value of interest.</td></tr> <tr> <td>N</td><td>The target table contains some subset of rows of primary key values.</td></tr> </table>	Y	The target table contains a row for every primary key value of interest.	N	The target table contains some subset of rows of primary key values.										
Y	The target table contains a row for every primary key value of interest.														
N	The target table contains some subset of rows of primary key values.														
TARGET_STRUCTURE	The structure of the target table: <table> <tr> <td>1</td><td>User table</td></tr> <tr> <td>3</td><td>CCD table</td></tr> <tr> <td>4</td><td>Point-in-time table</td></tr> <tr> <td>5</td><td>Base aggregate table</td></tr> <tr> <td>6</td><td>Change aggregate table</td></tr> <tr> <td>7</td><td>Replica</td></tr> <tr> <td>8</td><td>User copy</td></tr> </table>	1	User table	3	CCD table	4	Point-in-time table	5	Base aggregate table	6	Change aggregate table	7	Replica	8	User copy
1	User table														
3	CCD table														
4	Point-in-time table														
5	Base aggregate table														
6	Change aggregate table														
7	Replica														
8	User copy														

Table 82. Columns in the subscription member table (continued)

Column name	Description
PREDICATES	<p>Lists the predicates to be placed in a WHERE clause for the table in the TARGET_TABLE column. This WHERE clause creates a row subset of the source table. Predicates are recognized only when WHOS_ON_FIRST is set to S. The predicate cannot contain an ORDER BY clause because the Apply program cannot generate an ORDER BY clause. Aggregate tables require a dummy predicate followed by a GROUP BY clause.</p> <p>Because the Apply program uses these predicates for both full-refresh and change-capture replication, this column cannot contain predicates that involve columns in the CD or UOW table. Predicates that contain CD or UOW table references are stored in the UOW_CD_PREDICATES column.</p>
MEMBER_STATE	<p>A flag that indicates what state the member is in:</p> <p><b>N</b> (New) The member is new to this subscription set.</p> <p><b>L</b> (Loaded) The members of this subscription set have been loaded, but there has not yet been a change capture cycle.</p> <p><b>S</b> (Synchronized) The members of this subscription set are now synchronized up to a consistent synchpoint value.</p>
TARGET_KEY_CHG	<p>A flag that indicates how the Apply program handles updates when, at the source table, you change the source columns for the target key columns of a target table:</p> <p><b>Y</b> The Apply program updates the target table based on the before images of the target key column, meaning that the Apply program changes the predicate to the old values instead of the new. Make sure you have registered each before-image column of the target key so it is present in the CD table. For the corresponding registration entry in the register table, make sure the value in the CHG_UPD_TO_DEL_INS column is set to N.</p> <p><b>N</b> The Apply program uses logic while processing updates and deletes that assume that the columns that make up the target key are never updated.</p>

Table 82. Columns in the subscription member table (continued)

Column name	Description
JOIN_UOW_CD	<p>A flag that indicates whether the Apply program does a join of the CD and UOW tables when processing a user copy target table. This flag is needed when you define a subscription-set member with predicates that use columns from the UOW table that are not in the CD table. If the target table type is anything except user copy, then the Apply program uses a join of the CD and UOW tables when processing the member, and it ignores this column when processing the member.</p> <p><b>Y</b> The Apply program uses a join of the CD and UOW tables when processing the member.</p> <p><b>N</b> The Apply program does not use a join of the CD and UOW tables when processing the member; it reads changes only from the CD table.</p> <p><b>NULL</b> The Apply program ignores this column when processing the member. If the target table is a user copy and the value in this column is null, then the Apply program does not do a join of the CD and UOW tables when processing the member.</p>
UOW_CD_PREDICATES	<p>Contains predicates that include columns from the CD or UOW table that the Apply program needs only for change-capture replication, and not for full refreshes. During change-capture replication, the Apply program processes the predicates in this column and those in the PREDICATES column. During a full refresh, the Apply program processes only the predicates in the PREDICATES column.</p>
LOADX_TYPE	<p>The type of load for this member. The value in this column is used to override the defaults.</p> <p><b>NULL</b></p> <p><b>For z/OS:</b> The crossloader utility is used for this member.</p> <p><b>For UNIX and Windows:</b> The ASNLOAD exit determines the most appropriate utility for this member (option 3, 4, or 5).</p> <p><b>1</b> ASNLOAD is not used for this member. This effectively turns ASNLOAD option off for a particular subscription-set member even if you specified LOADX on startup.</p> <p><b>2</b> A user-defined or user-modified ASNLOAD exit code is used.</p> <p><b>3</b> The crossloader is used for this member.</p> <p><b>4</b> <b>For UNIX and Windows only:</b> EXPORT/LOAD is used for this member.</p> <p><b>5</b> <b>For UNIX and Windows only:</b> EXPORT/INPORT is used for this member.</p>

## ASN.IBMSNAP\_SUBS\_MEMBR

Table 82. Columns in the subscription member table (continued)

Column name	Description
LOAD_SRC_N_OWNER	The user-created nickname owner. This value is required if: <ul style="list-style-type: none"><li>• The crossloader is used for this member (LOADX_TYPE is 3).</li><li>• The target server is UNIX or Windows</li><li>• The source does not already reflect a nickname.</li></ul>
LOAD_SRC_N_TABLE	The user-created nickname table. This value is required if: <ul style="list-style-type: none"><li>• The crossloader is used for this member (LOADX_TYPE is 3).</li><li>• The target server is UNIX or Windows</li><li>• The source does not already reflect a nickname</li></ul>

## ASN.IBMSNAP\_SUBS\_SET

**Server:** Apply control server

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data.

The subscription sets table lists all of the subscription sets that are defined at the Apply control server and documents the replication progress for these sets. Rows are inserted into this table when you create your subscription set definition.

Table 83 provides a brief description of the columns in the subscription sets table.

Table 83. Columns in the subscription sets table

Column name	Description
APPLY_QUAL	Uniquely identifies which Apply program processes this subscription set.
SET_NAME	The name of the subscription set.
SET_TYPE	A flag that indicates whether the set is read only or read/write:  <b>R</b> The set is read only.  <b>U</b> The set is an update-anywhere configuration, and therefore is read/write.

Table 83. Columns in the subscription sets table (continued)

Column name	Description
WHOS_ON_FIRST	<p>The following values are used to control the order of processing in update-anywhere replication scenarios.</p> <p><b>F</b> (first) The source table is the replica and the target table is the master. In the case of update conflicts between the replica and the master table, the replica will have its conflicting transactions rejected. F is not used for read-only subscriptions; it is used for update anywhere.</p> <p><b>S</b> (second) The source table is the master table or other source, and the target table is the replica or other copy. In the case of update conflicts between the master and the replica table, the replica will have its conflicting transactions rejected. S is used for all read-only subscriptions.</p>
ACTIVATE	<p>A flag that indicates whether the Apply program will process the set during its next cycle:</p> <p><b>0</b> The subscription set is deactivated. The Apply program will not process the set.</p> <p><b>1</b> The subscription set is active indefinitely. The Apply program will process the set during each Apply cycle until you deactivate the set or until the Apply program is unable to process it.</p> <p><b>2</b> The subscription set is active for only one Apply cycle. The Apply program will process the set once and then deactivate the set.</p>
SOURCE_SERVER	The database name of the Capture control server where the source tables and views are defined.
SOURCE_ALIAS	The DB2 Universal Database alias corresponding to the Capture control server that is named in the SOURCE_SERVER column.
TARGET_SERVER	The database name of the server where target tables or views are stored.
TARGET_ALIAS	The DB2 Universal Database alias corresponding to the target server named in the TARGET_SERVER column.

Table 83. Columns in the subscription sets table (continued)

Column name	Description
STATUS	<p>A value that represents the work status for the Apply program after a given cycle:</p> <ul style="list-style-type: none"> <li>-1      The replication failed. The Apply program backed out the entire set of rows it had applied, and no data was committed. If the startup parameter <code>SQLERRCONTINUE = Y</code>, the <code>SQLSTATE</code> that is returned to the Apply program during the last cycle is <i>not</i> one of the acceptable errors you indicated in the input file for <code>SQLERRCONTINUE</code> (<i>apply_qualifier.SQS</i>).</li> <li>0      The Apply program processed the subscription set successfully. If the startup parameter <code>SQLERRCONTINUE = Y</code>, the Apply program did not encounter any SQL errors that you indicated for the <code>SQLERRCONTINUE</code> startup parameter (in <i>apply_qualifier.SQS</i>) and did not reject any rows.</li> <li>2      The Apply program is processing the subscription set in multiple cycles. It successfully processed a single logical subscription that was divided according to the <code>MAX_SYNC_MINUTES</code> control column.</li> <li>16     The Apply program processed the subscription set successfully and returned a status of 0; however, it encountered some SQL errors that you indicated for the <code>SQLERRCONTINUE</code> startup parameter (in <i>apply_qualifier.SQS</i>) and rejected some of the rows. See the <i>apply_qualifier.ERR</i> file for details about the rows that failed.</li> </ul> <p><b>Example:</b> You set <code>SQLERRCONTINUE = Y</code> and indicate that the allowable SQL state is 23502 (SQL code -407). A 23502 error occurs, but no other errors occur. The Apply program finishes processing the subscription set, and it sets the status to 16. On the next execution, a 23502 error occurs, but then a 07006 (SQL code -301) occurs. Now the Apply program stops processing the subscription set, backs out the entire set of rows it had applied, and sets the status to -1 (because no data was committed).</p> <ul style="list-style-type: none"> <li>18     The Apply program is processing the subscription set in multiple cycles and returned a status of 2, meaning that it successfully processed a single logical subscription that was divided according to the <code>MAX_SYNC_MINUTES</code> control column. However, it encountered some SQL errors that you indicated for the <code>SQLERRCONTINUE</code> startup parameter (in <i>apply_qualifier.SQS</i>) and rejected some of the rows. See the <i>apply_qualifier.ERR</i> file for details about the rows that failed.</li> </ul>

Table 83. Columns in the subscription sets table (continued)

Column name	Description
LASTRUN	The estimated time that the last subscription set began. The Apply program sets the LASTRUN value each time a subscription set is processed. It is the approximate time at the Apply control server when the Apply program begins processing the subscription set.
REFRESH_TYPE	<p>The type of scheduling that is used to prompt the Apply program to process this subscription set:</p> <p><b>R</b> The Apply program uses time-based scheduling. It uses the value in SLEEP_MINUTES to determine when to start processing the subscription set.</p> <p><b>E</b> The Apply program uses event-based scheduling. It checks the time value in the subscription events (IBMSNAP_SUBS_EVENT) table to determine when to start processing the subscription set. Before any replication (change capture or full refresh) can begin, an event must occur.</p> <p><b>B</b> The Apply program uses both time-based and event-based scheduling. Therefore, it processes the subscription set based on either the time or event criteria.</p>
SLEEP_MINUTES	Specifies the time (in minutes) of inactivity between subscription set processing. The processing time is used only when REFRESH_TYPE is R or B.
EVENT_NAME	A unique character string used to represent the name of an event. Use this identifier to update the subscription events table when you want to trigger replication for a subscription set. The event name is used only when REFRESH_TYPE is E or B.
LASTSUCCESS	The Apply control server timestamp for the beginning of the last successful processing of a subscription set.
SYNCHPOINT	The Apply program uses this column to record its progress, indicating that it has processed data up to this synchpoint value for the subscription set.
SYNCHTIME	The Apply program uses this column to record its progress, indicating that it has processed data up to this timestamp for the subscription set.
CAPTURE_SCHEMA	The schema name of the Capture control tables that process the source for this subscription set.
TGT_CAPTURE_SCHEMA	If the target table is also the source for another subscription set (such as an external CCD table in a multi-tier configuration or a replica table in an update-anywhere configuration), then this column contains the Capture schema that is used when the table is acting as a source.
FEDERATED_SRC_SRVR	The name of the federated remote server that is the source for the subscription set, which applies only to non-DB2 relational sources.

## ASN.IBMSNAP\_SUBS\_SET

Table 83. Columns in the subscription sets table (continued)

Column name	Description
FEDERATED_TGT_SRVR	The name of the federated remote server that is the target for the subscription set, which applies only to non-DB2 relational targets.
JRN_LIB	This column, which applies only to OS/400 Capture servers, is the library name of the journal that the source table uses.
JRN_NAME	This column, which applies only to OS/400 Capture servers, is the name of the journal used by a source table. An asterisk followed by nine blanks in this column means that the source table is currently not in a journal, in which case it is not possible to capture data for this source table.
OPTION_FLAGS	Reserved for future options of DB2 replication. Currently this column contains the default value of NNNN.
COMMIT_COUNT	<p>A flag that indicates the type of processing that the Apply program performs for a subscription set:</p> <p><b>NULL</b> This is the default setting for a read-only subscription set. The Apply program will process fetched answer sets for the <i>n</i> subscription-set members one member at a time, until all data has been processed, and then will issue a single commit at the end of the data processing for the whole set. The advantage of using this COMMIT_COUNT setting is that the processing might complete faster.</p> <p><i>Integer not NULL</i></p> <p>The Apply program processes the subscription set in a transactional mode. After all answer sets are fetched, the contents of the spill files will be applied in the order of commit sequence, ordering each transaction by the IBMSNAP_INTENTSEQ value order. This type of processing allows all spill files to be open and processed at the same time. A commit will be issued following the number of transactions specified in this column. For example, 1 means commit after each transaction, 2 means commit after each two transactions, and so on. An integer of 0 means that a single commit will be issued after all fetched data is applied. The advantage to using transactional mode processing is that the processing allows for referential integrity definitions at the target, and interim commits can be issued.</p>



Table 83. Columns in the subscription sets table (continued)

Column name	Description
MAX_SYNC_MINUTES	A time-threshold limit to regulate the amount of change data to fetch and apply during a subscription cycle. The Apply program breaks the subscription set processing into mini-cycles based on the IBMSNAP_LOGMARKER column in the UOW or CCD table at the Capture server and issues a COMMIT at the target server after each successful mini-cycle. The limit is automatically recalculated if the Apply program encounters a resource constraint that makes the set limit unfeasible. MAX_SYNC_MINUTES values that are less than 1 will be treated the same as a MAX_SYNC_MINUTES value equal to null.
AUX_STMTS	The number of SQL statements that you define in the subscription statement (IBMSNAP_SUBS_STMTS) table that can run before or after the Apply program processes a subscription set.
ARCH_LEVEL	The architectural level of the definition contained in the row. This column identifies the rules under which a row was created. This level is defined by IBM, and for Version 8 is 0801.

## ASN.IBMSNAP\_SUBS\_STMTS

**Server:** Apply control server

**Important:** Use caution when you update this table using SQL. Altering this table inappropriately can cause unexpected results and loss of data. The number of entries for a subscription should be reflected in the ASN.IBMSNAP\_SUBS\_SET.AUX\_STMTS column. If AUX\_STMTS is zero for a subscription set, the corresponding entries in the subscription statements table are ignored by the Apply program.

The subscription statements table contains the user-defined SQL statements or stored procedure calls that will be executed before or after each subscription-set processing cycle. Execute immediately (EI) statements or stored procedures can be executed at the source or target server only.

This table is populated when you define a subscription set that uses SQL statements or stored procedure calls.

Table 84 provides a brief description of the columns in the subscription statements table.

Table 84. Columns in the subscription statements table

Column name	Description
APPLY_QUAL	Uniquely identifies which Apply program processes the SQL statement or stored procedure.

Table 84. Columns in the subscription statements table (continued)

Column name	Description
SET_NAME	The name of the subscription set that the SQL statement or stored procedure is associated with.
WHOS_ON_FIRST	<p>The following values are used to control the order of processing in update-anywhere replication scenarios.</p> <p><b>F</b> (first) The target table is the user table or parent replica. The source table is the dependent replica and, in the case of update conflicts between the source table and the target table, the source table will have its conflicting transactions rejected. F is not used for read-only subscriptions.</p> <p><b>S</b> (second) The source table is the user table, parent replica, or other source. The target table is the dependent replica or other copy and, in the case of update conflicts between the source table and the target table, the target table will have its conflicting transactions rejected. S is used for all read-only subscriptions.</p>
BEFORE_OR_AFTER	<p>A value that indicates when and where the statement is issued:</p> <p><b>A</b> The statement is executed at the target server after all of the answer-set rows are applied.</p> <p><b>B</b> The statement is executed at the target server before any of the answer-set rows are applied.</p> <p><b>S</b> The statement is executed at the Capture control server before opening the answer-set cursors.</p> <p><b>G</b> Reserved for use by DB2 replication.</p> <p><b>X</b> Reserved for use by DB2 replication.</p>
STMT_NUMBER	Defines the relative order of execution within the scope of the BEFORE_OR_AFTER column value.
EI_OR_CALL	<p>A value that indicates:</p> <p><b>E</b> The SQL statement should be run as an EXEC SQL EXECUTE IMMEDIATE.</p> <p><b>C</b> The SQL statement contains a stored procedure name to run as an EXEC SQL CALL.</p>

Table 84. Columns in the subscription statements table (continued)

Column name	Description
SQL_STMT	One of the following values:  <b>Statement</b> The SQL statement should run as an EXEC SQL EXECUTE IMMEDIATE statement if EI_OR_CALL is E.  <b>Procedure</b> The eight-byte name of an SQL stored procedure, without parameters, or the CALL keyword that runs as an EXEC SQL CALL statement if EI_OR_CALL is C.
ACCEPT_SQLSTATES	One to ten five-byte SQLSTATE values that you specified when you defined the subscription set. These non-zero values are accepted by the Apply program as a successful execution. Any other values will cause a failed execution.

## Tables at the Monitor control server and their column descriptions

This section provides greater detail of each table stored at the Monitor control server. It also lists and briefly describes the columns in each table. The control tables are listed in alphabetical order, and the columns are listed in the order that they appear in each table from left to right.

### ASN.IBMSNAP\_ALERTS

**Server:** Monitor control server

The Monitor alerts table contains a record of all the alerts issued by the Replication Alert Monitor. It keeps track of what alert conditions occur, at which servers they occur, and when they were detected.

Table 85 provides a brief description of the columns in the Monitor alerts table.

Table 85. Columns in the Monitor alerts table

Column name	Description
MONITOR_QUAL	The Monitor qualifier identifying which Replication Alert Monitor program issued the alert.
ALERT_TIME	The time at the Monitor control server when the alert was inserted into this table.
COMPONENT	The replication component that is being monitored:  <b>C</b> Capture program <b>A</b> Apply program

## ASN.IBMSNAP\_ALERTS

Table 85. Columns in the Monitor alerts table (continued)

Column name	Description
SERVER_NAME	The name of the Capture control server or Apply control server where the alert condition occurred.
SERVER_ALIAS	The DB2 Universal Database alias of the Capture control server or Apply control server where the alert condition occurred.
SCHEMA_OR_QUAL	The Capture schema or Apply qualifier that is being monitored.
SET_NAME	If you set an alert condition for the Apply program, this column specifies the name of the subscription set that is being monitored. If you do not specify a set name, then monitoring is done at the Apply-qualifier level, meaning that every set within the given Apply qualifier is monitored.
CONDITION_NAME	The condition code that was tested when the alert was triggered.
OCCURRED_TIME	The time that the alert condition occurred at the Capture control or Apply control server.
ALERT_COUNTER	The number of times that this alert has been previously detected in consecutive monitor cycles.
ALERT_CODE	The message code that was issued when the alert occurred.
RETURN_CODE	The integer value returned by a user condition.
NOTIFICATION_SENT	A flag that indicates whether a notification message was sent:  Y      A notification message was sent.  N      A notification message was not sent because of the MAX_NOTIFICATIONS_PER_ALERT and MAX_NOTIFICATIONS_MINUTES parameters.
ALERT_MESSAGE	The text of the message that was sent, including the message code.

## ASN.IBMSNAP\_CONDITIONS

**Server:** Monitor control server

The Monitor conditions table contains the alert conditions for which the Replication Alert Monitor will contact someone, and it contains the group or individual's name to contact if a particular condition occurs. The Replication Alert Monitor can monitor a combination of conditions on Capture control and Apply control servers.

Table 86 on page 535 provides a brief description of the columns in the Monitor conditions table.

Table 86. Columns in the Monitor conditions table

Column name	Description
SERVER_NAME	The name of the Capture control or Apply control server where this condition is being monitored.
COMPONENT	The replication component that is being monitored:  <b>C</b> Capture program <b>A</b> Apply program
SCHEMA_OR_QUAL	The Capture schema or Apply qualifier that is being monitored.
SET_NAME	If you set an alert condition for the Apply program, this is the name of the subscription set you want monitored. If you do not specify a set name, the monitoring is done at the Apply qualifier level, meaning that every set within the Apply qualifier is monitored.
MONITOR_QUAL	The Monitor qualifier that identifies which Replication Alert Monitor program is monitoring the Capture control or Apply control server for this condition.
SERVER_ALIAS	The DB2 Universal Database alias of the Capture control server or Apply control server where this condition is being monitored.
ENABLED	A flag that indicates whether the Replication Alert Monitor will process this condition during the next monitoring cycle:  <b>Y</b> The Replication Alert Monitor will process this definition during the next cycle. <b>N</b> The Replication Alert Monitor will ignore this definition during the next cycle.
CONDITION_NAME	The name of the condition that the Replication Alert Monitor is monitoring at the given Capture control or Apply control server. Conditions for the Capture program begin with CAPTURE, and conditions for the Apply program begin with APPLY.  <b>CAPTURE_STATUS</b> The status of the Capture program. <b>CAPTURE_ERRORS</b> Whether the Capture program posted any error messages. <b>CAPTURE_WARNINGS</b> Whether the Capture program posts any warning messages.

## ASN.IBMSNAP\_CONDITIONS

Table 86. Columns in the Monitor conditions table (continued)

Column name	Description
CONDITION_NAME (Continued)	<p><b>CAPTURE_LASTCOMMIT</b> The last time the Capture program committed data during the last monitor cycle.</p> <p><b>CAPTURE_CLATENCY</b> The Capture program's current latency.</p> <p><b>CAPTURE_HLATENCY</b> Whether the Capture program's latency is greater than a certain number of seconds.</p> <p><b>CAPTURE_MEMORY</b> The amount of memory (in megabytes) that the Capture program is using.</p> <p><b>APPLY_STATUS</b> The status of the Apply program.</p> <p><b>APPLY_SUBSFAILING</b> Whether any subscription sets failed.</p> <p><b>APPLY_SUBSINACT</b> Whether any subscription sets either failed or are inactive.</p> <p><b>APPLY_ERRORS</b> Whether the Apply program posts any error messages.</p> <p><b>APPLY_WARNINGS</b> Whether the Apply program posts any warning messages.</p> <p><b>APPLY_FULLREFRESH</b> Whether a full refresh occurred.</p> <p><b>APPLY_REJTRANS (update anywhere)</b> Whether the Apply program rejects transactions in any subscription set.</p> <p><b>APPLY_SUBSDELAY</b> Whether the Apply program delays more than the time that you specified in the COND_INT parameter.</p> <p><b>APPLY_REWORKED</b> Whether the Apply program reworked any rows at the target table.</p> <p><b>APPLY_LATENCY</b> Whether the end-to-end latency of the Apply program exceeds a threshold.</p>
PARAM_INT	The integer parameter for the condition. The value of this parameter depends on the logic of the condition. It is passed as an input parameter whether it is used by the stored procedure or not.

Table 86. Columns in the Monitor conditions table (continued)

Column name	Description
PARM_CHAR	The character parameter for the condition. The value of this parameter depends on the logic of the condition. It is passed as an input parameter whether it is used by the stored procedure or not.
CONTACT_TYPE	A flag that indicates whether to contact an individual or a group if this condition occurs:  <div style="display: flex; justify-content: space-between; padding: 0 10px;"> <span><b>C</b></span> <span>Individual contact</span> </div> <div style="display: flex; justify-content: space-between; padding: 0 10px;"> <span><b>G</b></span> <span>Group of contacts</span> </div>
CONTACT	The name of the individual contact or the group of contacts to be notified if this condition occurs.

## ASN.IBMSNAP\_CONTACTGRP

**Server:** Monitor control server

The Monitor group contacts table contains the individual contacts that make up contact groups. You can specify for the Replication Alert Monitor to contact these groups of individuals if an alert condition occurs. An individual can belong to multiple contact groups (the columns are not unique).

Table 87 provides a brief description of the columns in the Monitor group contacts table.

Table 87. Columns in the Monitor group contacts table

Column name	Description
GROUP_NAME	The name of the contact group.
CONTACT_NAME	A contact name that is part of the group. These individuals are specified in the Monitor contacts (IBMSNAP_CONTACTS) table.

## ASN.IBMSNAP\_CONTACTS

**Server:** Monitor control server

The Monitor contacts table contains the necessary information for the Replication Alert Monitor to use to notify individuals when an alert condition that is associated with the individuals (or their group) occurs. One individual per row is specified.

Table 88 on page 538 provides a brief description of the columns in the Monitor contacts table.

## ASN.IBMSNAP\_CONTACTS

Table 88. Columns in the Monitor contacts table

Column name	Description
CONTACT_NAME	The name of the contact.
EMAIL_ADDRESS	The main e-mail or pager address for this contact.
ADDRESS_TYPE	A flag that indicates whether the e-mail address for this contact is an e-mail account or a pager address:  <b>E</b> The e-mail address is for an e-mail account. <b>P</b> The e-mail address is for a pager.
DELEGATE	The contact name to receive the notifications in a delegation period.
DELEGATE_START	The start date of a delegation period when notifications will be sent to individual named in the DELEGATE column.
DELEGATE_END	The end date of a delegation period.
DESCRIPTION	A description of the contact.

## ASN.IBMSNAP\_GROUPS

**Server:** Monitor control server

The Monitor groups table contains the name and description of each contact group. One group per row is specified.

Table 89 provides a brief description of the columns in the Monitor groups table.

Table 89. Columns in the Monitor groups table

Column name	Description
GROUP_NAME	The name of the contact group.
DESCRIPTION	A description of the contact group.

## ASN.IBMSNAP\_MONENQ

**Server:** Monitor control server

The Monitor enqueue table is reserved for future options of DB2 replication.

Table 90 provides a brief description of the columns in the Monitor enqueue table.

Table 90. Columns in the Monitor enqueue table

Column name	Description
MONITOR_QUAL	Reserved for future options of DB2 replication.



## ASN.IBMSNAP\_MONSERVERS

**Server:** Monitor control server

The Monitored servers table contains information about the last time that the Replication Alert Monitor monitored a Capture control or Apply control server.

Table 91 provides a brief description of the columns in the Monitored servers table.

*Table 91. Columns in the Monitored servers table*

Column name	Description						
MONITOR_QUAL	The Monitor qualifier that identifies which Replication Alert Monitor was monitoring the Capture control or Apply control server.						
SERVER_NAME	The name of the Capture control or Apply control server that the Replication Alert Monitor was monitoring.						
SERVER_ALIAS	The DB2 UDB alias of the Capture control or Apply control server that the Replication Alert Monitor was monitoring.						
LAST_MONITOR_TIME	The time (at the Capture control or Apply control server) that the Replication Alert Monitor program last connected to this server. This value is used as a lower bound value to fetch messages from the control tables and is the same value that START_MONITOR_TIME from the last successful monitor cycle.						
START_MONITOR_TIME	The time (at the Capture control or Apply control server) that the Replication Alert Monitor connected to the Capture control or Apply control server. This value is used as an upper bound value to fetch alert messages from the control tables.						
END_MONITOR_TIME	The time (at the Capture control or Apply control server) that the Replication Alert Monitor ended monitoring this server.						
LASTRUN	The last time (at the Monitor control server) when the Replication Alert Monitor started to process the Capture control or Apply control server.						
LASTSUCCESS	The value from the LASTRUN column of the last time (at the Monitor control server) when the Replication Alert Monitor successfully completed processing the Capture control or Apply control server. If the monitoring of this server keeps failing, the value could be the same (the history of this column is stored in the IBMSNAP_MONTRAIL table).						
STATUS	A flag that indicates the status of the monitoring cycle: <table> <tr> <td><b>-1</b></td><td>The Replication Alert Monitor failed to process this server successfully.</td></tr> <tr> <td><b>0</b></td><td>The Replication Alert Monitor processed this server successfully.</td></tr> <tr> <td><b>1</b></td><td>The Replication Alert Monitor is currently processing this server.</td></tr> </table>	<b>-1</b>	The Replication Alert Monitor failed to process this server successfully.	<b>0</b>	The Replication Alert Monitor processed this server successfully.	<b>1</b>	The Replication Alert Monitor is currently processing this server.
<b>-1</b>	The Replication Alert Monitor failed to process this server successfully.						
<b>0</b>	The Replication Alert Monitor processed this server successfully.						
<b>1</b>	The Replication Alert Monitor is currently processing this server.						

**ASN.IBMSNAP\_MONTRACE**

**Server:** Monitor control server

The Monitor trace table contains audit trail information for the Replication Alert Monitor. Everything that the Replication Alert Monitor does is recorded in this table, which makes it one of the best places for you to look if a problem with the Replication Alert Monitor program occurs.

Table 92 provides a brief description of the columns in the Monitor trace table.

*Table 92. Columns in the Monitor trace table*

Column name	Description
MONITOR_QUAL	The Monitor qualifier that identifies which Replication Alert Monitor issued the message.
TRACE_TIME	The timestamp when the message was inserted into this table.
OPERATION	A value used to classify messages:  <b>ERROR</b> An error message  <b>WARNING</b> A warning message  <b>INFO</b> An informational message
DESCRIPTION	The message code and text.

**ASN.IBMSNAP\_MONTRAIL**

**Server:** Monitor control server

The Monitor trail table contains information about each monitor cycle. The Replication Alert Monitor inserts one row for each Capture control or Apply control server that it monitors.

Table 93 provides a brief description of the columns in the Monitor trail table.

*Table 93. Columns in the Monitor trail table*

Column name	Description
MONITOR_QUAL	The Monitor qualifier that identifies which Replication Alert Monitor was monitoring the Capture control or Apply control server.
SERVER_NAME	The name of the Capture control or Apply control server that the Replication Alert Monitor was monitoring.
SERVER_ALIAS	The DB2 Universal Database alias of the Capture control or Apply control server that the Replication Alert Monitor was monitoring.

Table 93. Columns in the Monitor trail table (continued)

Column name	Description
STATUS	A flag that indicates the status of the monitoring cycle: <ul style="list-style-type: none"> <li><b>-1</b>      The Replication Alert Monitor failed to process this server successfully.</li> <li><b>0</b>        The Replication Alert Monitor processed this server successfully.</li> <li><b>1</b>        The Replication Alert Monitor is currently processing this server.</li> </ul>
LASTRUN	The time (at the Monitor control server) when the Replication Alert Monitor program last started to process the Capture control server or Apply control server.
LASTSUCCESS	The last time (at the Monitor control server) when the Replication Alert Monitor successfully completed processing the Capture control or Apply control server.
ENDTIME	The time when this row was inserted into this table.
LAST_MONITOR_TIME	The time (at the Capture control or Apply control server) when the Replication Alert Monitor last connected to the Capture control or Apply control server. This value is used as a lower bound value to fetch messages from the control tables and is the same value that START_MONITOR_TIME from the previous successful monitor cycle.
START_MONITOR_TIME	The last time when the Replication Alert Monitor started to monitor the Capture control or Apply control server.
END_MONITOR_TIME	The last time when the Replication Alert Monitor finished monitoring the Capture control or Apply control server.
SQLCODE	The SQLCODE of any errors that occurred during this monitor cycle.
SQLSTATE	The SQLSTATE of any errors that occurred during this monitor cycle.
NUM_ALERTS	The number of alert conditions that occurred during this monitor cycle.
NUM_NOTIFICATIONS	The number of notifications that were sent during this monitor cycle.

### Tables at the target server and their column descriptions

This section provides greater detail of each table used by the target server. It also lists and briefly describes the columns in each table. The table names are listed in alphabetical order, and the column names are listed in the order that they appear in each table from left to right.

#### Base aggregate table

*schema.base\_aggregate*

**Server:** target server

## Base aggregate table

**Important:** If you use SQL to update this table, you run the risk of losing your updates if a full refresh is performed by the Apply program.

A base aggregate table is a target table that contains the results of aggregate functions that are performed on data located at the source table.

Table 94 provides a brief description of the columns in the base aggregate table.

*Table 94. Columns in the base aggregate table*

Column name	Description
<i>user columns</i>	The aggregate data that was computed from the source table.
IBMSNAP_LLOGMARKER	The current timestamp at the source server when the aggregation of the data in the source table began.
IBMSNAP_HLOGMARKER	The current timestamp at the source server when the aggregation of the data in the source table completed.

## Change aggregate table

*schema.change\_aggregate*

**Server:** target server

**Important:** If you use SQL to update this table, you run the risk of losing your updates if a full refresh is performed by the Apply program.

A change aggregate table is a target table that contains the results of aggregate functions that are performed on data in the change-data (CD) table. This table is similar to the base aggregate table, except that the functions being performed at the CD table are done only for changes that occur during a specific time interval.

Table 95 provides a brief description of the columns in the change aggregate table.

*Table 95. Columns in the change aggregate table*

Column name	Description
<i>user key columns</i>	The columns that make up the target key.
<i>user nonkey columns</i>	The nonkey data columns from the source table. The column names in this target table do not need to match the column names in the source table, but the data types must match.
<i>user computed columns</i>	User-defined columns that are derived from SQL expressions. You can use computed columns with SQL functions to convert source data types to different target data types.

Table 95. Columns in the change aggregate table (continued)

Column name	Description
IBMSNAP_LLOGMARKER	The oldest IBMSNAP_LOGMARKER or IBMSNAP_LLOGMARKER value present in the (CD+UOW) or CCD table rows being aggregated.
IBMSNAP_HLOGMARKER	The newest IBMSNAP_LOGMARKER or IBMSNAP_HLOGMARKER value present in the (CD+UOW) or CCD table rows being aggregated.

## Consistent-change data (CCD) table

*schema.CCD\_table*

This table contains information that you can update by using SQL.

**Server:** target server

**Important:** If you use SQL to update this table, you run the risk of losing your updates if a full refresh is performed by the Apply program.

Consistent-change-data (CCD) tables are targets in subscription-set members that contain information about changes that occur at the source, and have additional columns to identify the sequential ordering of those changes. The values in the columns come from a join of the CD and UOW tables. A CCD table that is at the target server can be:

- An internal CCD table that acts as an alternative to the CD table.  
For change-capture replication, the Apply program applies changes to the targets directly from this table. The name of this type of CCD table is stored in the same row in the register (IBMSNAP\_REGISTER) table as the replication source that it holds changes from.
- An external CCD that is a read-only target table.  
This type of CCD contains a audit trail of your source data at the target server.
- An external CCD that is a middle tier in a multi-tier replication configuration.  
This type of CCD is a target table for tier 1 and a source table for tier 3. The name of this type of CCD table is stored in its own row in the register (IBMSNAP\_REGISTER) table.

For more information on the uses for CCD tables as targets, see “Selecting a target type” on page 78.

The Capture program does not insert data into CCD tables and does not prune them. Instead, your application requirements should determine the history retention period for CCD tables. Therefore, pruning of CCD tables is

# Consistent-change data table

not automatic by default, but can be easily automated using an SQL statement to be processed after the subscription cycle.

For external CCDs, you can choose to include several columns from the UOW table: APPLY\_QUAL, IBMSNAP\_AUTHID, IBMSNAP\_AUTHTKN, IBMSNAP\_REJ\_CODE, and IBMSNAP\_UOWID.

The originally captured operation code in the IBMSNAP\_OPERATION column and the sequence numbers IBMSNAP\_INTENTSEQ and IBMSNAP\_COMMITSEQ are included in CCD tables. For condensed CCD tables, only the latest values are kept for each row.

For information on CCD tables that are populated by Capture triggers or that contain non-relational data, see “*schema.CCD\_table (non-DB2)*” on page 491.

Table 96 provides a brief description of the columns in the CCD table.

Table 96. Columns in the CCD table

Column name	Description
IBMSNAP_INTENTSEQ	The log or journal record sequence number that uniquely identifies a change. This value is globally ascending.
IBMSNAP_OPERATION	A flag indicating the type of operation for a record. <div><div>I</div>Insert <div>U</div>Update <div>D</div>Delete</div>
IBMSNAP_COMMITSEQ	The log record sequence number of the captured commit statement. This value groups inserts, updates, and deletes by the original transactions for the source table.
IBMSNAP_LOGMARKER	The commit time at the Capture control server.
<i>user key columns</i>	If the CCD table is condensed, this column contains the columns that make up the target key.
<i>user nonkey columns</i>	The nonkey data columns from the source table. The column names in this target table do not need to match the column names in the source table, but the data types must be compatible.
<i>user computed columns</i>	User-defined columns that are derived from SQL expressions. You can use computed columns with SQL functions to convert source data types to different target data types.
APPLY_QUAL (optional)	Uniquely identifies which Apply program processes this CCD table.

Table 96. Columns in the CCD table (continued)

Column name	Description										
IBMSNAP_AUTHID (optional)	The authorization ID associated with the transaction. It is useful for database auditing. AUTHID length is 18 characters. If you supply a longer value, it is truncated. For DB2 Universal Database for z/OS, this column is the primary authorization ID. For DB2 Universal Database for iSeries, this column has the name of the user profile ID under which the application that caused the transaction ran. This column holds a 10-character ID padded with blanks. This column is not automatically copied to other tables; you must select it and copy it as a user data column. This column can be selected as a user data column for a noncomplete CCD target table.										
IBMSNAP_AUTHTKN (optional)	The authorization token associated with the transaction. This ID is useful for database auditing. For DB2 Universal Database for z/OS, this column is the correlation ID. For DB2 Universal Database for iSeries, this column is the job name of the job that caused a transaction. This column is not automatically copied to other tables; you must select it and copy it as a user data column. This column can be selected as a user data column for a noncomplete CCD target table.										
IBMSNAP_REJ_CODE (optional)	<p>This value is set only during update-anywhere replication if conflict detection is specified as standard or advanced when you define your replication source. It is not valid for non-DB2 relational targets because they cannot participate in update-anywhere configurations.</p> <table> <tr> <td><b>0</b></td><td>A transaction with no known conflict.</td></tr> <tr> <td><b>1</b></td><td>A transaction that contains a conflict where the same row in the source and replica tables have a change that was not replicated. When a conflict occurs, the transaction will be reversed at the replica table.</td></tr> <tr> <td><b>2</b></td><td>A cascade-rejection of a transaction dependent on a prior transaction having at least one same-row conflict. When a conflict occurs, the transaction will be reversed at the replica table.</td></tr> <tr> <td><b>3</b></td><td>A transaction that contains at least one referential-integrity constraint violation. Because this transaction violates the referential constraints defined on the source table, the Apply program will mark this subscription set as failed. Updates cannot be copied until the referential integrity definitions are corrected.</td></tr> <tr> <td><b>4</b></td><td>A cascade-rejection of a transaction dependent on a prior transaction having at least one constraint conflict.</td></tr> </table>	<b>0</b>	A transaction with no known conflict.	<b>1</b>	A transaction that contains a conflict where the same row in the source and replica tables have a change that was not replicated. When a conflict occurs, the transaction will be reversed at the replica table.	<b>2</b>	A cascade-rejection of a transaction dependent on a prior transaction having at least one same-row conflict. When a conflict occurs, the transaction will be reversed at the replica table.	<b>3</b>	A transaction that contains at least one referential-integrity constraint violation. Because this transaction violates the referential constraints defined on the source table, the Apply program will mark this subscription set as failed. Updates cannot be copied until the referential integrity definitions are corrected.	<b>4</b>	A cascade-rejection of a transaction dependent on a prior transaction having at least one constraint conflict.
<b>0</b>	A transaction with no known conflict.										
<b>1</b>	A transaction that contains a conflict where the same row in the source and replica tables have a change that was not replicated. When a conflict occurs, the transaction will be reversed at the replica table.										
<b>2</b>	A cascade-rejection of a transaction dependent on a prior transaction having at least one same-row conflict. When a conflict occurs, the transaction will be reversed at the replica table.										
<b>3</b>	A transaction that contains at least one referential-integrity constraint violation. Because this transaction violates the referential constraints defined on the source table, the Apply program will mark this subscription set as failed. Updates cannot be copied until the referential integrity definitions are corrected.										
<b>4</b>	A cascade-rejection of a transaction dependent on a prior transaction having at least one constraint conflict.										
IBMSNAP_UOWID (optional)	The unit-of-work identifier from the log record header for this unit of work.										

**Related reference:**

## Consistent-change data table

- “*schema.CCD\_table* (non-DB2)” on page 491

### Point-in-time table

*schema.point\_in\_time*

**Server:** target server

**Important:** If you use SQL to update this table, you run the risk of losing your updates if a full refresh is performed by the Apply program.

The point-in-time table contains a copy of the source data, with an additional system column (IBMSNAP\_LOGMARKER) containing the timestamp of approximately when the particular row was inserted or updated at the source server.

Table 97 provides a brief description of the columns in the point-in-time table.

*Table 97. Columns in the point-in-time table*

Column name	Description
<i>user key columns</i>	The columns that make up the target key.
<i>user nonkey columns</i>	The nonkey data columns from the source table or view. The column names in this target table do not need to match the column names in the source table, but the data types must match.
<i>user computed columns</i>	User-defined columns that are derived from SQL expressions. You can use computed columns with SQL functions to convert source data types to different target data types.
IBMSNAP_LOGMARKER	The approximate commit time at the Capture control server. This column is null following a full refresh.

### Replica table

*schema.replica*

**Server:** target server

This table contains information that you can update by using SQL.

The replica table must have the same key columns as the source table. Because of these similarities, the replica table can be used as a source table for further subscription sets. Converting a target table into a source table is done automatically when you define a replica target type and specify the CHANGE DATA CAPTURE attribute. See “Defining read-write targets (update-anywhere)” on page 87 for more information.

Table 98 on page 547 provides a brief description of the columns in the replica table.



Table 98. Columns in the replica table

Column name	Description
<i>user key columns</i>	The columns that make up the target key, which must be the same primary key as the master table.
<i>user nonkey columns</i>	The nonkey data columns from the source table. The column names in this target table do not need to match the column names in the source table, but the data types must match.

## User copy table

*schema.user\_copy*

**Server:** target server

**Important:** If you use SQL to update this table, you run the risk of losing your updates if a full refresh is performed by the Apply program.

The user copy table is a target table that contains a copy of the columns in the source table. This target table can be a row or column subset of the source table, but it cannot contain any additional columns.

Except for subsetting and data enhancement, a user copy table reflects a valid state of the source table, but not necessarily the most current state. References to user copy tables (or any other type of target table) reduce the risk of contention problems that results from a high volume of direct access to the source tables. Accessing local user copy tables is much faster than using the network to access remote source tables for each query.

Table 99 provides a brief description of the columns in the user copy table.

Table 99. Columns in the user copy table

Column name	Description
<i>user key columns</i>	The columns that make up the target key.
<i>user nonkey columns</i>	The nonkey data columns from the source table or view. The column names in this target table do not need to match the column names in the source table, but the data types must match.
<i>user computed columns</i>	User-defined columns that are derived from SQL expressions. You can use computed columns with SQL functions to convert source data types to different target data types.

## User copy table

---

## Chapter 24. Replication messages

This chapter contains a list of the messages issued by DB2 replication for the replication programs on all of the database management systems except DB2 for iSeries. Following each message in this chapter you will find an explanation for the message and a suggestion for the action you might take to resume your operations. You can also obtain explanations for messages by typing the following command at a DB2 command prompt:

*db2 message\_number*

**Note for iSeries:** The replication messages issued by DB2 for iSeries are available online only, from the library of replication message files (MSGF). To display a message from the command line, type [DSPMSGD RANGE(message\_number) MSGF(library\_name/filename)].

The replication messages in this chapter have the following ranges:

**ASN0001 to ASN0499**

Messages for the Capture program.

**ASN0500 to ASN0999**

Messages common to the Capture and Apply programs.

**ASN1000 to ASN1499**

Messages for the Apply program.

**ASN1500 to ASN1999**

Messages for the Replication Center.

**ASN5000 to ASN5099**

Messages associated with migration.

**ASN5100 to ASN5199**

Messages for the Replication Alert Monitor.

**ASN5200 to ASN5299**

Messages for NT Services.

Unless otherwise stated, all error codes in this chapter are internal error codes used by IBM Software Support to locate the code where the particular message was issued. Also, unless otherwise stated, error messages are issued with a nonzero return code.

**Tips for Capture errors:** If you receive a Capture program error, verify that your DB2 maintenance is at the most current level. The Capture program is an application program that uses DB2 APIs. Many Capture program errors are due to DB2 maintenance that is not current.

---

**ASN0004E** CAPTURE *capture\_schema*. The Capture program could not start the trace. The return code is *return\_code*. The reason code is *reason\_code*.

**Explanation:** An error occurred when the START TRACE DB2 command was issued, or when the Capture program read the DB2 log.

**User Response:** See the DB2 Codes section in the messages and codes documentation of the DB2 database manager on your operating system to find the appropriate reason code. For more information, see either of the following administration documentation: the Call Attachment Facility (CAF) for START TRACE DB2 errors, or the Instrumentation Facility Interface (IFI) for DB2 log read errors, or contact your DBA. If the CAF or the IFI returned a message, it is also printed on the system display console.

---

**ASN0005E** CAPTURE *capture\_schema*. The Capture program encountered an error when reading the DB2 log. The log sequence number is *lsn*, the SQLCODE is *sql\_return\_code*, and the reason code is *reason\_code*.

**Explanation:** An error occurred when the Capture program read the DB2 log. There might be an SQL error.

- For DB2 Replication, the *sqlcode* value is for the Asynchronous Read Log API.
- For Capture for VSE, the *sqlcode* is for the VSE/VSAM GET macro.
- For Capture for VM, the *sqlcode* is for Diagnose X'A4'.

**User Response:** See the DB2 Codes section in the messages and codes documentation of the DB2 database manager on your operating system for the appropriate reason code, as suggested below:

- For Capture program for z/OS, see the Instrumentation Facility Interface (IFI) section in the administration documentation of the DB2 database manager on your operating system, or contact your DBA.

- For Capture for VSE, see the VSE/VSAM Commands and Macros, VSE/ESA System Macro Reference, and VSE/ESA V2R3 Messages and Codes manuals for more information.
- For VM/ESA, see the VM/ESA Programming Services for more information.
- For Capture on Linux, Windows, and UNIX, see the active and archived database logs administration documentation for DB2 Universal Database, or contact IBM Software Support.

---

**ASN0006E** CAPTURE *capture\_schema*. The Capture program encountered an unexpected log error of unknown log variation.

**Explanation:** An unexpected log error occurred when the Capture program was processing the DB2 log records, and was not reported by either of the following interfaces:

- The Instrumentation Facility Interface (IFI) for Capture program for z/OS
- The Asynchronous Read Log API for the Capture program

The Capture program could not determine which type of SQL update was associated with the log record.

**User Response:** Contact IBM Software Support.

---

**ASN0008I** CAPTURE *capture\_schema*. The Capture program was stopped.

**Explanation:** The Capture program has stopped.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0009E** CAPTURE *capture\_schema*. The registered source table *src\_owner.src\_table* does not have the DATA CAPTURE CHANGES attribute.

**Explanation:** When attempting to initialize a registration, the Capture program encountered a source table that is incorrectly defined. The

Capture program cannot process the log records associated with a source table if the DATA CAPTURE CHANGES attribute of the source table is not set. This registration is not valid and typically occurs only if you generated this registration outside of the supplied administration tools. This message is issued during a Capture program warm start or when the first CAPSTART signal is received for a subscription set against this registration. If this error occurs during the processing of a CAPSTART signal, the Capture program does not activate the registration. If this error occurs during a reinitialization (from a warm start or a reinit capture command), the Capture program places the registration in the "stopped" state, indicating that you must repair the registration before the Apply program can resynchronize the associated subscription sets.

**User Response:**

1. Alter the source table to turn on data capture changes. For example:
  - *alter table regress.table3 data capture changes*
2. If the registration has been deactivated by the Capture program (state = stopped), update the registration to set the state to inactive.
3. Use the Replication Center to force the Apply program to perform a full refresh for all subscription sets that replicate from this source table.

---

**ASN0011E** CAPTURE *capture\_schema*. The Capture program log read failed because the DB2 compression dictionary that was used to create the compressed log record no longer exists. The log record that could not be read was for the registered source table *src\_owner.src\_table*. The reason code is *reason\_code*.

**Explanation:** The Capture program received a nonzero response code from the DB2 log read IFI. The response code indicates that the data on a log record cannot be processed because the compression dictionary for the corresponding DB2 table space is not available.

The compressed table space containing this source table was probably reorganized by the REORG utility that ran without the KEEPDICTIONARY option. The Capture program must deactivate this registration, because the remaining compressed log records cannot be read. The Capture program cannot continue unless this registration is deactivated or removed. This error does not cause the Capture program to terminate.

**User Response:** See the Maintaining your replication environment chapter for restrictions about compressed table spaces and for more information about deactivated registrations and corresponding full refreshes by the Apply programs.

---

**ASN0013E** CAPTURE *capture\_schema*. The Capture program required a column that was not defined in the change data (CD) table. The table name is *table\_name*.

**Explanation:** A required column in the change data table is not defined.

**User Response:** Ensure that the change data table definition is correct. Refer to the Tables structures documentation in the DB2 Replication Guide and Reference for more information.

---

**ASN0019E** CAPTURE *capture\_schema*. The Capture program libraries are not authorized for the Authorized Program Facility (APF).

**Explanation:** The Capture program cannot start.

**User Response:** Authorize the Capture link library for APF, and restart the program.

---

**ASN0020I** CAPTURE *capture\_schema*. Netview Generic Alerts Interface failure. The Netview return code is *return\_code*.

**Explanation:** The Network Major Vector Transport (NMVT) could not be sent to Netview by the program because the program interface

failed. This is a secondary informational message.

**User Response:** See the Netview programming documentation for a description of the return code to determine the interface error. The Capture program alerts are not received by the System Services Control Point (SSCP) until the error is corrected.

---

**ASN0021I**    **CAPTURE** *capture\_schema*. **Netview Program to Program Interface unavailable. The Netview return code is** *return\_code*.

**Explanation:** Netview is unavailable. This is a secondary informational message.

**User Response:** See the Netview programming documentation for a description of the return code to determine the Netview problem. For example, the subsystem might not have been started.

---

**ASN0023I**    **CAPTURE** *capture\_schema*. **The Capture program has been reinitialized and is capturing changes for** *number* **registrations. Stopped** *number* **registrations are in a stopped state. Inactive** *number* **registrations are in an inactive state.**

**Explanation:** A REINIT command was issued to the Capture program. The Capture program then tried to refresh all the internal control information for all the registrations.

**User Response:** If the Capture program is capturing changes for all the registrations, no action is required. Otherwise, examine the preceding error messages to determine the cause of the failure, and follow the suggested user response to repair the failing registration definition. After you repair the registration, re-issue the REINIT command to the Capture program.

---

**ASN0028I**    **CAPTURE** *capture\_schema*. **The Capture program is suspended by an operator command.**

**Explanation:** An operator command has suspended the Capture program, and the program has entered a wait state.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0029I**    **CAPTURE** *capture\_schema*. **The Capture program is resumed by an operator command.**

**Explanation:** An operator command has resumed the Capture program from a suspended state, and the Capture program has continued running.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0031E**    **CAPTURE** *capture\_schema*. **The program parameter table IBMSNAP\_CAPPARMS can have only one row.**

**Explanation:** The Capture program parameters table was not defined correctly or has been updated with rows that are not valid.

**User Response:** Make sure that there is only one row in the IBMSNAP\_CAPPARMS table. Refer to the Table structures documentation in the DB2 Replication Guide and Reference for additional information.

---

**ASN0035E**    **CAPTURE** *capture\_schema*. **A row with an unsupported architecture level was found in the table IBMSNAP\_REGISTER. The row is not valid and specifies CD table** *cd\_owner.cd\_table*, **and the architecture level is** *arch\_level*.

**Explanation:** The Capture program tried to initialize a registration and found that the registration definition contains an architecture level that is not valid. The Version 8 Capture program can use only registrations that are at the

architecture level of *0801*. This registration is not valid and typically occurs if you generated this registration without using the Replication Center. This message is issued during Capture warm start or when the first CAPSTART signal is received for a subscription against this registration. This error does not cause the Capture program to terminate.

**User Response:** Refer to the Tables structures documentation in the DB2 Replication Guide and Reference to check the required value for the ARCH\_LEVEL column in the register table. Verify that the value in the register table at the source server is correct. If the value is not correct, update the value of the architectural level of the registration and use compatible versions of the Replication Center and the Capture program.

---

**ASN0049I**    **CAPTURE** *capture\_schema*. **A row for the SIGNAL\_SUBTYPE CAPSTOP was inserted into the table IBMSNAP\_SIGNAL.**

**Explanation:** The Capture program received a signal to stop capturing data. The Capture program commits current work in progress and terminates.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0055E**    **CAPTURE** *capture\_schema*. **The Capture program encountered a column *column\_name* with an unsupported SQLTYPE in table *table\_name*.**

**Explanation:** The Capture program tried to initialize a registration and found that the registration definition contains an unsupported SQL type such as an abstract type. This registration is not valid and typically occurs if you generate this registration outside of the supplied administration tools. This message is issued during Capture warm start or when the first CAPSTART signal is received for a subscription against this registration. This error does not cause the Capture program to terminate.

**User Response:** Remove this registration; it cannot be supported by DB2 replication.

---

**ASN0057E**    **CAPTURE** *capture\_schema*. **The Capture program encountered error *errno* on operation for file *filename*.**

**Explanation:** An error occurred when the Capture program was handling files. The Capture program terminates.

**User Response:** Ensure that the Capture program has correct access and security permissions for all required paths and files. Also, ensure that adequate space is available on your system. If you believe that this message was issued because of product failure, contact IBM Software Support for assistance.

---

**ASN0058W**    **CAPTURE** *capture\_schema*. **The MAP\_ID *mapid* in a CAPSTART row in the IBMSNAP\_SIGNAL table does not correspond to any entry in the IBMSNAP\_PRUNCNTL table.**

**Explanation:** The value for the MAP\_ID that is specified by the CAPSTART signal does not match any current value in the MAP\_ID column of the IBMSNAP\_PRUNCNTL table. The subscription set might have been deleted, or a user might have inserted the CAPSTART signal incorrectly.

**User Response:** If this CAPSTART was issued by a user, check that the MAP\_ID for the Signal table insert is correct, and try again. If this CAPSTART signal was issued by the Apply program, verify that the subscription set still exists.

---

**ASN0059W**    **CAPTURE** *capture\_schema*. **The SYNCHPOINT field in the IBMSNAP\_PRUNCNTL table is not zeros for the CAPSTART of subscription with MAP\_ID *map\_id*.**

**Explanation:** When the Apply program signals a full refresh to the Capture program, the Apply

program inserts a row for the CAPSTART signal in the IBMSNAP\_SIGNAL table. At the same time, the SYNCHPOINT column of the IBMSNAP\_PRUNCNTL table is set to hex zeroes. The Capture program then responds to the Apply program to confirm that the Capture program received the CAPSTART signal, as follows: the Capture program sets the value for the SYNCHPOINT column in the IBMSNAP\_PRUNCNTL table to the number of the log sequence that corresponds to the CAPSTART log record. Because the Apply program set the value in the SYNCHPOINT column to hex zeroes, the Apply program checks if a nonzero value has been inserted by the Capture program. The Capture program updates the value for SYNCHPOINT, even if the value was not hex zeroes. However, if the value for SYNCHPOINT is not hex zeroes, the Capture program issues this warning that the value it found was not expected.

This warning can occur if you issue the APPLY CAPSTART signal yourself and do not completely simulate the actions of the Apply program.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0060E**    **CAPTURE** *capture\_schema*. The Capture program encountered an internal error *error\_code*.

**Explanation:** An unexpected error occurred in the Capture program. The Capture program terminates.

**User Response:** Contact IBM Software Support for assistance.

---

**ASN0061E**    **CAPTURE** *capture\_schema*. A registration that is not valid has been found. Source table *source\_owner.source\_table* does not exist in the system catalog tables.

**Explanation:** The Capture program tried to initialize a registration and found that the source table specified in the registration was not in the source system catalog. This message is issued

during Capture warm start or when the Apply program issues the first CAPSTART signal for a subscription set that contains a registration that is not valid. This error does not cause the Capture program to terminate. The values for the columns *source\_owner* and *source\_table* in the Capture control table IBMSNAP\_REGISTER might have been incorrectly specified, or the source table was dropped and no longer exists.

**User Response:** If the registration is in error, correct the values for the columns *source\_owner* and *source\_table*. If the source table no longer exists, then the registration is no longer valid and can be removed.

---

**ASN0062E**    **CAPTURE** *capture\_schema*. A registration that is not valid has been found. CD table *phys\_change\_owner.phys\_change\_table* does not exist in the system catalog tables.

**Explanation:** The Capture program tried to initialize a registration and found that the CD table specified in the registration was not in the source system catalog tables. This message is issued during Capture warm start or when the first CAPSTART signal is received for a subscription against this registration. This error does not cause the Capture program to terminate. The values for the columns *phys\_change\_owner* and *phys\_change\_table* in the Capture control table IBMSNAP\_REGISTER might have been incorrectly specified, or the CD table was dropped and no longer exists.

**User Response:** If the registration is in error, correct the values in the columns *phys\_change\_owner* and *phys\_change\_table*. If the CD table no longer exists, then the registration is no longer valid and can be removed.



---

**ASN0063E**    **CAPTURE** *capture\_schema*. The source table *source\_owner.source\_table* associated with the subscription having *MAP\_ID mapid* does not exist in the system catalog tables. The Capture program could not start capturing changes for this subscription.

**Explanation:** The Capture program tried to respond to a CAPSTART signal for a subscription and found that the source table which corresponds to the subscription was not in the source system catalog tables. This error message is issued when the first CAPSTART signal is received for a subscription that is not valid. This error does not cause the Capture program to terminate. The values for the columns *source\_owner* and *source\_table* in the Capture control table IBMSNAP\_PRUNCNTL might have been incorrectly specified, or the source table was dropped and no longer exists.

**User Response:** If the subscription is in error, correct the values for the columns *source\_owner* and *source\_table*. If the source table no longer exists, then the subscription is no longer valid and can be removed.

---

**ASN0064E**    **CAPTURE** *capture\_schema*. The registration is not valid for an associated subscription having *MAP\_ID mapid*. The Capture program cannot start capturing change data for this subscription.

**Explanation:** The Capture program tried to initialize a registration associated with a particular subscription and found that the registration contains one or more column values that are not valid. This message is issued when the first CAPSTART signal for a subscription is received against this registration. This error does not cause the Capture program to terminate. The values for the columns *phys\_change\_owner* and *phys\_change\_table* in the Capture control table IBMSNAP\_REGISTER might have been incorrectly specified, or the CD table was dropped and no longer exists.

**User Response:** If the registration is in error, correct the values for the columns *phys\_change\_owner* and *phys\_change\_table*. If the registration is no longer needed, you can remove it.

---

**ASN0065E**    **CAPTURE** *capture\_schema*. A registration that is not valid has been found. The source table *source\_owner.source\_table* is not a local physical table.

**Explanation:** The Capture program tried to initialize a registration and found that the source table for the registration is a not a local physical table, but is instead on a non-DB2 relational server that is used as a source and called by a nickname. When a non-DB2 relational server is used as a source, data from each server is captured through a trigger program. Each source table must be in its own register table built on the non-DB2 relational server. This message is issued during a Capture warm start or when the first CAPSTART signal against this registration is received for a subscription. This error does not cause the Capture program to terminate. The registration has been created incorrectly in an IBMSNAP\_REGISTER table in a DB2 database.

**User Response:** This registration must be rebuilt and made valid at the correct non-DB2 relational server.

---

**ASN0066E**    **CAPTURE** *capture\_schema*. A registration that is not valid has been found. The CD table *phys\_change\_owner.phys\_change\_table* is not a local physical table.

**Explanation:** The Capture program tried to initialize a registration and found that the CD table for the registration that corresponds to the subscription is a nickname for a non-DB2 relational database used as a source. This message is issued when the first CAPSTART signal is received for a subscription against this registration. This error does not cause the Capture program to terminate.

**User Response:** Non DB2 relational sources are captured through trigger programs, and must be

in their own register table in the non DB2 relational source system. CCD tables for such sources are also created in the non DB2 relational source system. Somehow the non-DB2 relational source table registration has been incorrectly registered in a register table in a DB2 database. This registration must be rebuilt at the correct non-DB2 relational server.

---

**ASN0067E**    **CAPTURE** *capture\_schema*. The view registration associated with the subscription having MAP\_ID *map\_id* was not found in the table IBMSNAP\_REGISTER. The Capture program could not start capturing change data for this subscription.

**Explanation:** The Capture program tried to initialize a registration and found that the view registration that corresponds to the subscription does not exist. This message is issued during Capture warm start or when the first CAPSTART signal is received for a subscription against this registration. This error does not cause the Capture program to terminate. The values of the columns *source\_owner*, *source\_table*, and *source\_view\_qual* in the IBMSNAP\_REGISTER or IBMSNAP\_PRUNCNTL Capture control table could have been incorrectly specified. Therefore, either no match was found or the registration was dropped and no longer exists.

**User Response:** If the subscription or the registration is in error, correct the values in the columns *source\_owner*, *source\_table*, and *source\_view\_qual*. If the registration no longer exists, then the subscription is no longer valid and can be removed.

---

**ASN0068E**    **CAPTURE** *capture\_schema*. The insert statement is too long for CD table  
*phys\_chg\_owner.phys\_chg\_tbl*.

**Explanation:** The number of columns in the CD table is too large; the SQL INSERT statement exceeds the 32K Capture coding limit.

**User Response:** If all of the table columns are defined in the registration but only a subset of

these columns are needed at the target, reduce the number of columns for that registration. Alternatively, split the table over two registrations so that each registration has a different subset of the table columns.

---

**ASN0069E**    **CAPTURE** *capture\_schema*.  
SQLCODE *sqlcode* was returned during an insert into the CD table  
*phys\_chg\_owner.phys\_chg\_tbl*. The CD table appears to have been dropped.

**Explanation:** The Capture program tried to insert a row into a CD table, and DB2 returned a SQLCODE indicating that the CD table no longer exists. The CD table could inadvertently have been dropped, or the whole registration could have been dropped. If there are still rows in the IBMSNAP\_REGISTER table that refer to this CD table, the Capture program deactivates these registrations by setting the value of the CD\_OLD\_SYNCHPOINT column to NULL and no longer attempts to capture changes for this CD table. This error does not cause the Capture program to terminate.

**User Response:** If the CD table no longer exists and is no longer required, the registration is no longer valid and should be removed. It is recommended that you deactivate the registration before you remove it. Any subscription sets associated with registrations that use this CD table should also be deactivated. Additionally, the associated subscription-set members should be removed so that these subscription sets can be activated and can run successfully.

---

**ASN0070E** CAPTURE *capture\_schema*. The combination of column name *column\_name* in the CD table *phys\_chg\_owner.phys\_chg\_tbl*, and the value of **BEFORE\_IMG\_PREFIX** *before\_img\_prefix* in the table **IBMSNAP\_REGISTER** for this registration matches multiple column names in the source table. The ambiguity in the registration definition cannot be resolved by the Capture program.

**Explanation:** The Capture program tried to initialize a registration and found that a column within the CD table for the registration is ambiguous. The column could refer to either a before image for one source column or to an after image for another source column. This message is issued during Capture warm start or when the first CAPSTART signal is received for a subscription against this registration. This error does not cause the Capture program to terminate.

**User Response:** In the **IBMSNAP\_REGISTER** table, change the current value in the **BEFORE\_IMG\_PREFIX** column to a character value that does not produce this ambiguity.

---

**ASN0071E** CAPTURE *capture\_schema*. The data type attribute of the column *column\_name* in the CD table *phys\_chg\_owner.phys\_chg\_tbl* is not compatible with the data type attribute of the corresponding source column.

**Explanation:** The Capture program tried to initialize a registration and found that a column within the CD table for the registration is not compatible with the corresponding source column. This message is issued during Capture warm start or when the first CAPSTART signal is received for a subscription against this registration. This error does not cause the Capture program to terminate.

**User Response:** Correct the CD table for this registration.

---

**ASN0072E** CAPTURE *capture\_schema*. The before-image column *column\_name* in the CD table *phys\_chg\_owner.phys\_chg\_tbl* must allow NULL values.

**Explanation:** The Capture program tried to initialize a registration and found that the before-image column within the CD table for the registration was not defined to accept null values. This message is issued during Capture warm start or when the first CAPSTART signal is received for a subscription against this registration. This error does not cause the Capture program to terminate.

**User Response:** Correct the CD table for this registration.

---

**ASN0073E** CAPTURE *capture\_schema*. The specification *input\_in* in describing the CD table on a CAPSTOP signal is not valid.

**Explanation:** The Capture program found that the **INPUT\_IN** value specified on the CAPSTOP signal is not in a valid format of *phys\_change\_owner.phys\_change\_table*. This error does not cause the Capture program to terminate, and no action is taken for this signal.

**User Response:** Ensure that the value of **INPUT\_IN** matches the name of the CD table associated with the registration that you want to deactivate. Insert a new row into the **IBMSNAP\_SIGNAL** table.

---

**ASN0074E** CAPTURE *capture\_schema*. There is no row in the **IBMSNAP\_REGISTER** table that corresponds to *src\_owner.src\_table* that is specified on a CAPSTOP signal.

**Explanation:** The Capture program found that the **INPUT\_IN** value specified on the CAPSTOP signal is in a valid format, and there is no match for the value of *source\_owner.source\_table* in the registration table. This error does not cause the Capture program to terminate.

**User Response:** Correct the value of INPUT\_IN, and insert the signal again.

---

**ASN0075W** **CAPTURE** *capture\_schema*. The registration corresponding to the INPUT\_IN, *src\_owner.src\_table* on a CAPSTOP signal was not capturing changes. No action is taken.

**Explanation:** The Capture program found that the INPUT\_IN value specified on the CAPSTOP signal is in a valid format and matches the value of a *source\_owner.source\_table* in the registration table, but this registration is already inactive. This error does not cause the Capture program to terminate, and the Capture program takes no action for the signal.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0076I** **CAPTURE** *capture\_schema*. Capture has stopped capturing changes for source table *source\_owner.source\_table* in response to a CAPSTOP signal.

**Explanation:** The Capture program successfully deactivated a registration that was specified in a CAPSTOP signal.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0077E** **CAPTURE** *capture\_schema*. The values specified for the PHYS\_CHANGE\_OWNER and PHYS\_CHANGE\_TABLE columns in the IBMSNAP\_PRUNCNTL table where MAP\_ID = *mapid* are NULL or do not match a valid row in the IBMSNAP\_REGISTER table.

**Explanation:** The Capture program tried to initialize a registration and found that the column values of PHYS\_CHANGE\_OWNER and PHYS\_CHANGE\_TABLE within the IBMSNAP\_PRUNCNTL table for the subscription do not match a registration row in the

IBMSNAP\_REGISTER table. The message was issued during Capture warm start or when the first CAPSTART signal was received for a subscription against this registration. This error does not cause the Capture program to terminate.

**User Response:** Correct the values in the IBMSNAP\_PRUNCNTL table for this subscription. If this subscription was built using the Replication Center, contact IBM Software Support to report a potential administration problem.

---

**ASN0078E** **CAPTURE** *capture\_schema*. The before-image column *column\_name* in the CD table *phys\_owner.phys\_table* has no corresponding after-image column in the CD table for this registration. The registration is not valid.

**Explanation:** The Capture program tried to initialize a registration and found a before-image column within the CD table of the registration that has no corresponding after-image column. This registration is not valid and typically occurs only if you generated this registration outside of the supplied administration tools. This message is issued during a Capture program warm start or when the first CAPSTART signal is received for a subscription against this registration. This error does not cause the Capture program to terminate.

**User Response:** Correct the CD table for this registration by ensuring that before-image columns are included in the CD table only when the corresponding after-image column is also included.

---

**ASN0079E** **CAPTURE** *capture\_schema.*  
**SQLCODE** *sqlcode* **was returned**  
**during an update to the**  
**IBMSNAP\_REGISTER table for**  
**the registrations associated with**  
**the CD table**  
*phys\_chg\_owner.phys\_chg\_tbl. The*  
**rows might have been deleted.**

**Explanation:** The Capture program tried to update the IBMSNAP\_REGISTER table to indicate that data has been captured for the named CD table, and DB2 returned a SQLCODE indicating that the rows no longer exist. The registrations could have been dropped. This error does not cause the Capture program to terminate.

**User Response:** If the registration has been dropped, no further action is required for the registrations. When dropping registrations, it is recommended that you deactivate the registrations first. If the rows in the IBMSNAP\_REGISTER table were inadvertently deleted, drop the associated CD table and rebuild the registrations. Deactivate any subscription sets that are associated with the registrations. If a registrations must be dropped, remove the associated subscription-set members so that these subscription sets can be activated and can run successfully. If the registrations are rebuilt, a signal is sent to the Apply program indicating that a full refresh should be performed for the associated subscription sets.

---

**ASN0080E** **CAPTURE** *capture\_schema.* **A table**  
**space full condition has been**  
**encountered for CD table**  
*phys\_chg\_owner.phys\_chg\_tbl, which*  
**is associated with the registration**  
**for source table**  
*source\_owner.source\_table.*

**Explanation:** The Capture program tried to process an insert into the named CD table but was unable to process the insert due to a table space full condition. Typically this condition results from insufficient space allocation for CD table spaces, infrequent pruning, or ineffective pruning. This error causes the Capture program to terminate.

**User Response:** Take the following steps to determine the cause of the table space full condition:

1. Ensure that sufficient space has been allocated to the table space for this CD table in order to accommodate normal processing conditions.
2. Ensure that pruning is performed often enough to reduce the storage requirements for the Capture control tables.
3. Ensure that the Apply programs are running often enough to accommodate normal pruning processing.
4. Verify that no subscription sets have been deactivated for a long period of time without taking the additional steps necessary for normal pruning.

Refer to the DB2 Replication Guide and Reference for additional information.

---

**ASN0082W** **CAPTURE** *capture\_schema.* **The**  
**Capture program encountered a**  
**registration with a column**  
**column\_name in the CD table**  
*phys\_chg\_owner.phys\_chg\_tbl with a*  
**column length** *CD\_column\_length*  
**that is shorter than the length of**  
**the corresponding column in the**  
**source table**  
*source\_owner.source\_table, with a*  
**length of** *src\_column\_length.*

**Explanation:** During the initialization of a registration, the Capture program found that the registration definition contains a column in the CD table with a column length that is shorter than the corresponding column length in the source table. This is not a recommended registration definition and typically occurs only if you generated or altered this registration outside of the supplied administration tools. The registration definition is allowed, but a warning message is issued to inform you that the captured source table data might not fit within the defined CD table column. This message is issued during a Capture program warm start or when the first CAPSTART signal is received for a subscription against this registration. The

registration initializes successfully.

**User Response:** Unless there is a specific reason why you need to define the registration in this manner (for example, if you are certain that the length of the changed data will never be larger than the length of the CD table column), you should define the registration so that the source table and the CD table data definitions match exactly.

---

**ASN0083E** **CAPTURE** *capture\_schema*. **SQLCODE** *sqlcode* **was returned when trying to process an insert into the CD table**  
*phys\_chg\_owner.phys\_chg\_tbl*. **The CD table column** *column\_name* **is too short and cannot contain the captured data from the corresponding column in the source table,**  
*source\_owner.source\_table*. **The registration has been stopped by the Capture program.**

**Explanation:** The Capture program tried to process an insert into a CD table and encountered a SQLCODE from DB2 that indicates that the CD table contains a column that is shorter than the length of the corresponding column in the source table. This is not a recommended registration definition and typically occurs only if you generated or altered this registration outside of the supplied administration tools. This error does not cause the Capture program to terminate, but the registration is placed in the stopped state.

**User Response:** Re-evaluate this registration definition. Either alter the registration so that the lengths of the source table column and the CD table column match or add a trigger to the CD table to truncate the data.

---

**ASN0084E** **CAPTURE** *capture\_schema*. **The registration with the source table** *source\_owner.source\_table* **and the CD table**  
*phys\_chg\_owner.phys\_chg\_tbl* **has been stopped by the Capture program.**

**Explanation:** This error message is issued any time that a registration is placed in the stopped state (with the STATE column set to a value of 'S' in the IBMSNAP\_REGISTER table) by the Capture program. The reason for this action is described in one or more of the preceding messages.

**User Response:** Examine the preceding error messages to determine the cause of the failure, and follow the suggested user response to repair the failing registration definition. After you repair the registration definition, you must manually set the value of the STATE column to 'T' in the IBMSNAP\_REGISTER table to indicate that the registration can be used again by the Apply program.

---

**ASN0100I** **CAPTURE** *capture\_schema*. **The Capture program initialization is successful.**

**Explanation:** This message is for your information only.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0101W** **CAPTURE** *capture\_schema*. **The Capture program warm start failed because existing data is too old. A cold start will be attempted.**

**Explanation:** The data in the change data tables is too old. A cold start is performed.

**User Response:** See the documentation regarding Capture program operations in the DB2 Replication Guide and Reference.

---

**ASN0102W**    **CAPTURE** *capture\_schema*. The Capture program switches to cold start because the warm start information is insufficient.

**Explanation:** A problem occurred during the retrieval of the restart information. The restart table data is not valid. A cold start is performed.

- For DB2 Universal Database, an Asynchronous Read Log API error occurred during warm start while DB2 was reading the log.
- For z/OS, an Instrumentation Facility Information (IFI) error occurred during warm start while DB2 was reading the log.

**User Response:** See the documentation regarding Capture program operations in the DB2 Replication Guide and Reference.

---

**ASN0104I**    **CAPTURE** *capture\_schema*. In response to a CAPSTART signal with MAP\_ID *mapid*, change capture has started for the source table *source\_owner.source\_table* for changes found on the log beginning with log sequence number *log\_sequence\_number*.

**Explanation:** The Capture program successfully processed a CAPSTART signal. If this is the first CAPSTART signal associated with a particular source table, this message indicates that the Capture program is now capturing updates to the source table. If this is a CAPSTART signal for a table for which changes are already being captured, this message indicates that the Capture program received the signal and performed the required processing to allow the Apply program to start receiving changes for the subscription set that is associated with the input MAP\_ID value.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0105I**    **CAPTURE** *capture\_schema*. *n* rows have been pruned from the table *table\_owner.table\_name* at timestamp.

**Explanation:** The Capture program pruned records from a CD, UOW, TRACE, MONITOR, or SIGNAL table.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0109I**    **CAPTURE** *capture\_schema*. The Capture program has successfully initialized and is capturing data changes for *number* registrations. Stopped *number* registrations are in a stopped state. Inactive *number* registrations are in an inactive state.

**Explanation:** This message is issued when the Capture program completes the reinitialization of registration entries. The reinitialization can occur during a warm start, during the processing of a CAPSTART signal, or in response to a Capture REINIT command.

**User Response:** If the Capture program is capturing the changes for all the registrations, no action is required. Otherwise, examine the preceding error messages to determine the cause of the failure, and follow the suggested user responses to repair the failing registration definition. After you have repaired the registration definition, issue the `asncmd` command with the `reinit` parameter.

---

**ASN0111I**    **CAPTURE** *capture\_schema*. The pruning cycle started at timestamp.

**Explanation:** This message is issued at the beginning of each pruning cycle.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0112I**    **CAPTURE** *capture\_schema*. The pruning cycle ended at timestamp.

**Explanation:** This message is issued at the termination of each pruning cycle.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0113W** **CAPTURE** *capture\_schema*. The pruning retention limit will be reached in the next 24 hours.

**Explanation:** This warning message is issued when the values in the IBMSNAP\_UOW table indicate that retention limit pruning could occur in the next day.

**User Response:** Check to see why regular pruning is not occurring. Usually this is because one or more Apply programs have not been run for a period of many days, and therefore the CD and UOW tables cannot be effectively pruned. Another potential hazard is the removal or deactivation of a subscription set, without the removal or reset of the corresponding synchpoint value in the table IBMSNAP\_PRUNE\_SET. The Replication Analyzer tool can be used to provide a detailed analysis of the situation.

---

**ASN0114E** **CAPTURE** *capture\_schema*. Pruning has failed with SQL code *sqlcode* when pruning the table *table\_owner.table\_name*.

**Explanation:** This error message is issued when pruning fails with an unexpected SQL error code. Pruning terminates and tries again at the next interval or command invocation. This error does not cause the Capture program to terminate.

**User Response:** If this SQL code indicates a temporary error, then no action is required. Otherwise, take action as indicated for the SQL error in the DB2 Messages and Codes manual.

---

**ASN0121E** **CAPTURE** *capture\_schema*. The Capture program warm start failed because existing data is too old. The Capture program will terminate.

**Explanation:** The time of the warm start information exceeded the lag limit.

**User Response:** No response required; the Capture program will terminate.

---

**ASN0122E** **CAPTURE** *capture\_schema*. An error occurred while reading the restart information or DB2 log. The Capture program will terminate.

**Explanation:** A problem occurred while retrieving the restart information. The restart table data was not valid or for z/OS, an Instrumentation Facility Interface (IFI) error occurred while reading the log during a restart. When the error is resolved, you can restart using the warm start option.

**User Response:** No response required; the Capture program terminates.

---

**ASN0123I** **CAPTURE** *capture\_schema*. At program termination, the highest log sequence number of a successfully captured log record is *max\_commitseq* and the lowest log sequence number of a record still to be committed is *min\_inflightseq*.

**Explanation:** The Capture program terminates and records the values of the restart table at that time for auditing purposes.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0133I** **CAPTURE** *capture\_schema*. The Capture program has reached the end of the active log and will terminate because the AUTOSTOP feature is specified.

**Explanation:** The Capture program terminated when it reached the end of the active log as requested by the user option of AUTOSTOP.

**User Response:** This message is for your information only, and no action is required.



---

**ASN0142E** **CAPTURE** *capture\_schema*. The Capture program is unable to perform an insert operation on the monitor table **IBMSNAP\_CAPMON**. The SQL code is *sqlcode*. The monitoring information for this interval will be skipped.

**Explanation:** This error message is issued when the monitoring thread has failed with an unexpected SQL code. Monitor functions for this interval are skipped, and the program tries again at the next interval. This error does not cause the Capture program to terminate.

**User Response:** If this SQL code indicates a temporary error, then no action is required. Otherwise, take action as indicated for the SQL error in the DB2 Messages and Codes manual.

---

**ASN0143W** **CAPTURE** *capture\_schema*. The program detected that the source database *src\_db\_name* has been restored or rolled forward. The Capture program has switched from a warm start to a cold start.

**Explanation:** The Capture program started with a startmode of *warmsa* or *warmsi*. When the Capture program attempted to warm start, it received a return code from the DB2 log read API that indicates that the source database has been restored or rolled forward and that log sequence numbers have been reused; the state of the source database and the state of the captured data are no longer consistent. The Capture program switched to a cold start.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0144E** **CAPTURE** *capture\_schema*. The program detected that the source database *src\_db\_name* has been restored or rolled forward. A cold start is recommended to restore consistency.

**Explanation:** The Capture program started with a startmode of *warmns* or *warmsi*. When the

Capture program attempted to warm start, it received a return code from the DB2 log read API that indicates that the source database has been restored or rolled forward and that log sequence numbers have been reused; the state of the source database and the state of the captured data are no longer consistent. The Capture program terminates and does not automatically switch to a cold start.

**User Response:** If you are certain that it is safe to perform a Capture program warm start, restart the Capture program; it will not terminate on a second attempt. If you are not certain whether the captured data will be in a consistent state after a Capture program warm start, it is recommended that you perform a Capture program cold start.

---

**ASN0180W** **CAPTURE** *capture\_schema*. The table **IBMSNAP\_SIGNAL** is not an EBCDIC table as required by capture. The signal has been processed.

**Explanation:** The Capture program detected that the **IBMSNAP\_SIGNAL** table is not defined as an EBCDIC table. Additional processing is required to translate signals to EBCDIC to process them properly. The additional processing requires a small performance degradation.

**User Response:** At your earliest convenience, perform the following steps:

1. Stop the Capture program.
2. Drop and re-create the **IBMSNAP\_SIGNAL** table with EBCDIC encoding.
3. Restart the Capture program.

---

**ASN0181W** **CAPTURE** *capture\_schema*. The row for the signal with timestamp *signal\_time* no longer exists in the **IBMSNAP\_SIGNAL** table. The signal has been processed.

**Explanation:** The Capture program processed the request from the signal but could not update the **SIGNAL\_STATE** and **SIGNAL\_LSN**. Therefore, the issuer of the signal cannot

determine that the Capture program received the signal.

**User Response:** Determine if another process is expecting the update for the signal from the Capture program, and if necessary, re-send the signal.

---

**ASN0182W** *CAPTURE capture\_schema. The row for signal with timestamp signal\_time no longer exists in IBMSNAP\_SIGNAL table and the table is not EBCDIC. The signal will be ignored by capture.*

**Explanation:** An initialization failure occurred, because the Capture program received a signal that was not encoded in EBCDIC. The Capture program could not translate the signal to EBCDIC, because the row in the IBMSNAP\_SIGNAL table no longer exists. The Capture program cannot determine what signal was sent and so ignores it.

**User Response:** Determine what signal was sent, and re-send the signal.

At your earliest convenience, perform the following steps:

1. Stop the Capture program.
2. Drop and re-create the IBMSNAP\_SIGNAL table with EBCDIC encoding.
3. Restart the Capture program.

---

**ASN0500E** *pgmname : program\_qualifier : The parameter input input\_value supplied for parameter name parameter\_name is not valid.*

**Explanation:** The program or a command program has been invoked with a specified input parameter that is not valid. The message indicates the name of the program that is reporting the error, along with the parameter name and the parameter value.

**User Response:** Check the documentation on valid invocation parameters, correct the input, and resubmit the task or command.

---

**ASN0501E** *pgmname : program\_qualifier : The value input\_value supplied for the parameter parameter\_name is not the correct data type.*

**Explanation:** The program or a command program was invoked with an input value with an associated data type that is not valid. The message indicates the name of the program that is reporting the error, the incorrect input value, and the name of the parameter for which this input value was specified.

**User Response:** Correct the invocation to include the correct data type for the parameter input and resubmit it.

---

**ASN0502E** *pgmname : program\_qualifier : The value input\_value of length invalid\_string\_length, supplied for parameter parameter\_name, is greater than the maximum allowed string length of allowed\_string\_length.*

**Explanation:** The program or a command program has been invoked using an input value with a string length that is not valid. The message indicates the name of the program that is reporting the error, what input value is incorrect, and for which parameter this input value was specified.

**User Response:** Correct the invocation to include the correct string length for the parameter input and resubmit it.

---

**ASN0503E** *pgmname : program\_qualifier : The integer value input\_value, supplied for parameter parameter\_name, is outside the supported range for this parameter.*

**Explanation:** The program or a command program was invoked with an input value specified which is outside the supported range. The message indicates the name of the program that is reporting the error, which input value is incorrect, and for which parameter this input value was specified.

**User Response:** Correct the invocation to include the correct range value for the parameter input and resubmit it.

---

**ASN0504E** *pgmname : program\_qualifier : The program did not recognize the invocation parameter incorrect\_input.*

**Explanation:** The program or a command program has been invoked with a specified parameter or command that is not valid. The message indicates which program issued this message, and the invocation input that is unrecognized.

**User Response:** Check the documentation on valid input parameters, correct the input and resubmit the task or command.

---

**ASN0505E** *pgmname : program\_qualifier : The program was unable to get or set an IPC key.*

**Explanation:** The program or a command program was unable to initialize the inter-process communications needed to process commands. This error causes the failing program to terminate.

**User Response:** Retry the failing program or command. Contact IBM Software Support if the problem persists.

---

**ASN0506E** *pgmname : program\_qualifier : The program could not attach to the replication communications message queue.*

**Explanation:** The program or a command program encountered an internal error while trying to process a user command. The program did not terminate for this failure, but the command did not get executed.

**User Response:** Retry the failing command. Contact IBM Software Support if the problem persists.

---

**ASN0507E** *pgmname : program\_qualifier : The program could not create the replication communications message queue.*

**Explanation:** The program or a command program encountered an internal error while trying to process a user command. The program did not terminate for this failure, but the command did not get executed

**User Response:** Retry the failing command. Contact IBM Software Support if the problem persists.

---

**ASN0508E** *pgmname : program\_qualifier : The program could not send a message to the replication communications message queue.*

**Explanation:** The program or a command program encountered an internal error while trying to process a user command. The program did not terminate for this failure, but the command did not get executed.

**User Response:** Retry the failing command. Contact IBM Software Support if the problem persists.

---

**ASN0509E** *pgmname : program\_qualifier : The program could not process a received message because of an incorrect message version.*

**Explanation:** The program or a command program encountered an internal error while trying to process a user command. The program did not terminate for this failure, but the command did not get executed.

**User Response:** Retry the failing command. Contact IBM Software Support if the problem persists.

---

**ASN0510E** *pgmname : program\_qualifier : The program encountered a timeout while waiting for reply messages.*

**Explanation:** The command program encountered an internal error while trying to

process a user command. The program did not terminate for this failure, but the command did not get executed.

**User Response:** Retry the failing command. Contact IBM Software Support if the problem persists.

---

**ASN0511E** *pgmname : program\_qualifier : The program was unable to process the received message because of an unknown message function.*

**Explanation:** The program encountered an internal error while trying to process a user command. The program did not terminate for this failure, but the command did not get executed.

**User Response:** Retry the failing command. Contact IBM Software Support if the problem persists.

---

**ASN0512E** *pgmname : program\_qualifier : The program could not read from its replication communications message queue.*

**Explanation:** The program encountered an internal error while trying to process a user command. The program did not terminate for this failure, but the command did not get executed.

**User Response:** Retry the failing command. Contact IBM Software Support if the problem persists.

---

**ASN0513E** *pgmname : program\_qualifier : The program could not open the message file named msg\_file.*

**Explanation:** This message file used by the program has been installed incorrectly, or the language environment variables are not set correctly.

**User Response:** Refer to the documentation for information about installation and configuration.

---

**ASN0514E** *pgmname : program\_qualifier : The program could not open the log file log\_file.*

**Explanation:** The program encountered an internal error while trying to open a file for its own program message log, and terminates abnormally because of this failure. This problem might have occurred because the file was inadvertently deleted, or because the userid associated with this process does not have the sufficient authority to open the file.

**User Response:** Verify that sufficient authority is provided to the processing userid. If the file was inadvertently deleted, restart the program to create a new log file.

---

**ASN0515E** *pgmname : program\_qualifier : The program could not close the log file.*

**Explanation:** The program encountered an internal error while trying to close the file used for its own program message log. The file might have been deleted inadvertently before the program tried to terminate. Final termination messages might not be issued.

**User Response:** If the file was inadvertently deleted, restart the program to create a new log file.

---

**ASN0516E** *pgmname : program\_qualifier : The program could not close the message catalog.*

**Explanation:** The program encountered an internal error while trying to close the message catalog file. The file might have been deleted inadvertently before the program tried to terminate. Final termination messages might not be issued.

**User Response:** If the message file has been deleted, it needs to be reinstalled.

---

**ASN0517E**    *pgmname : program\_qualifier : The program has recovered the ability to read from its replication communications message queue.*

**Explanation:** The program was able to reinitialize its read capability from the message queue needed to process commands after a previous failure.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0518E**    *pgmname : program\_qualifier : The program does not accept multiple commands.*

**Explanation:** The command program was invoked with multiple commands specified. Each command invocation must be performed with a single input command, along with any other required command input. Note: The CHGPARMs command allows multiple parameters to be changed with one invocation of the CHGPARMs command.

**User Response:** Correct the command input, and resubmit the command.

---

**ASN0519E**    *pgmname : program\_qualifier : The parameter input parameter\_value supplied for CHGPARMs parameter parameter\_name is not valid.*

**Explanation:** The CHGPARMs command was invoked with incorrect parameter input.

**User Response:** Correct the command input and resubmit the command.

---

**ASN0520I**    *pgmname : program\_qualifier : The STATUS command response: thread\_type thread is in the status\_condition state.*

**Explanation:** In response to the **status** command, one of these messages will be issued for each of the threads associated with the program that received the command, in each case providing the current state of that thread.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0521I**    *pgmname : program\_qualifier : The QRYPARMS command response: parameter\_name was set to parameter\_value by the following method: method.*

**Explanation:** In response to the **QRYPARMS** command, a message will be issued for each of the program parameters. For each parameter, the message provides the name of the parameter, the current setting of the parameter, and the method (by default, by changing the IBMSNAP\_CAPPARMs table, by the startup option, or by the use of the CHGPARMs command) that was employed by the user to set the value of the parameter.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0522I**    *pgmname : program\_qualifier : The program received the command\_type command.*

**Explanation:** The program received a command to be processed.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0523I**    *pgmname : program\_qualifier : The CHGPARMs command response: parameter\_name has been set to parameter\_value.*

**Explanation:** In response to the **CHGPARMs** command, one of these messages will be issued for each of the program parameters that was changed. For each parameter, the message provides the new value for the parameter.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0524E** *pgmname : program\_qualifier : The program required parameter parameter\_name was not specified.*

**Explanation:** The program or a command program was invoked without one of the required parameters specified. If the parameter missing is the *capture\_server* or *control\_server*, the program or command also tried accessing the database name implicitly through the DB2DBDFT environment variable setting, where applicable, and this also was not successful.

**User Response:** Correct the invocation to include the appropriate parameter and its corresponding input value.

---

**ASN0525E** *pgmname : program\_qualifier : The program could not read from its external communications message queue.*

**Explanation:** The program or a command program was unable to initialize its read capability from the external communications message queue needed to process commands.

**User Response:** Retry the failing command, and if the problem persists, contact IBM Software Support.

---

**ASN0526E** *pgmname : program\_qualifier : The program was invoked without any command input.*

**Explanation:** The command program was invoked without a command to process. No command processing is performed.

**User Response:** Resubmit the command with all the required input.

---

**ASN0527E** *pgmname : program\_qualifier : The program was invoked without any CHGPARMS command input.*

**Explanation:** The commands program was invoked with the CHGPARMS command but without any command input to process. No command processing is performed.

**User Response:** Resubmit the command with all the required input.

---

**ASN0528E** *pgmname : program\_qualifier : The program will terminate because the required control table control\_table\_name does not exist.*

**Explanation:** The Capture or the Apply program tried to execute an SQL operation against a required Capture control table. The program received a *not found* return code from DB2. The return code occurs either if the migration has not been completed or if a required Capture control table has been accidentally dropped from the environment.

**User Response:** See the message text for the name of the missing control table. Corrective action for this problem depends on which table is missing. For example, if the table is IBMSNAP\_PRUNE\_LOCK, then the table can simply be recreated, and the Capture program can be restarted. However, if the table is IBMSNAP\_RESTART, and if the correct table contents cannot be restored, then the table needs to be recreated, and the Capture program requires a cold start.

---

**ASN0529I** *pgmname : program\_qualifier : The value of parameter\_name was set to parameter\_value at startup by the following method: method.*

**Explanation:** The program started, and the program parameters were initialized based on the combination of startup options specified and the existing contents of the table IBMSNAP\_CAPPARMS. These messages were written for auditing purposes. The parameters were set by one of the methods: by default, by changing the IBMSNAP\_CAPPARMS table, or by the startup option.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0530E** *pgmname : program\_qualifier : The program could not connect to database database\_name with USERID user\_id . The SQLCODE is sql\_code.*

**Explanation:** An error occurred when the program issued one of the following functions:

- a CONNECT function to DB2 for VSE and VM
- a CONNECT function to DB2 Call Attachment Facility (CAF)
- an implicit connect to DB2 UDB

**User Response:** See the DB2 codes in the messages and codes publication of the DB2 database manager on your operating system for the appropriate reason code.

For DB2 for z/OS errors, see the section in the administration guide that describes the Call Attachment Facility. Contact your DBA for questions and diagnosis.

If you are running the program under DB2 UDB for UNIX, ensure that the LIBPATH environment variable is set to the same environment in which the program starts. Refer to the Setting up for replication documentation in the DB2 Replication Guide and Reference for additional information.

---

**ASN0531E** *pgmname : program\_qualifier : The program could not open the plan. The SQL return code is return\_code, the reason code is reason\_code, the subsystem name is DB2\_subsystem, and the plan name is plan\_name.*

**Explanation:** An error occurred when the program tried to open the plan, ASNLPLAN.

**User Response:** See the DB2 Codes section in the messages and codes publication of the DB2 database manager on your operating system to find the appropriate reason code. See the section in the administration guide that describes the Call Attachment Facility.

---

**ASN0532E** *pgmname : program\_qualifier : DB2 release release\_number is not supported.*

**Explanation:** The program does not support this release of DB2.

**User Response:** Run the program with the appropriate release of DB2.

---

**ASN0533E** *pgmname : program\_qualifier : DB2 was terminated abnormally.*

**Explanation:** DB2 was terminated while the program was still active.

For z/OS, VSE/ESA, or VM/ESA, DB2 was terminated while program was active and the user did not specify the NOTERM invocation parameter.

**User Response:** Start DB2 and start the program.

---

**ASN0534E** *pgmname : program\_qualifier : DB2 database cannot be used, because it is in the state state.*

**Explanation:** DB2 was terminated while the program was still active. The database is in one of the following states: UNDETERMINED, TERMINATED, QUIESCED, ROLLWARD, or ACTIVE.

**User Response:** Start DB2, and then start the program.

---

**ASN0535E** *pgmname : program\_qualifier : The program could not disconnect from the database db\_server. The return code is return\_code, and the reason code is reason\_code.*

**Explanation:** While terminating the connection to DB2, the program received an error code from the Call Attachment Facility (CAF).

**User Response:** Restart the program.

---

**ASN0536E** *pgmname : program\_qualifier : An error was returned while getting the instance name. The SQLCODE is sqlcode.*

**Explanation:** The SQLEGENS API of DB2 Universal Database returned an error.

**User Response:** See the DB2 Universal Database API Reference for information about the SQLEGENS API to determine the error, or contact IBM Software Support.

---

**ASN0537E** *pgmname : program\_qualifier : The program could not connect to database database\_name, the return code is return\_code, and the reason code is reason\_code.*

**Explanation:** An error occurred when the program issued one of the following functions:

- a CONNECT function to DB2 for VSE and VM
- a CONNECT function to DB2 Call Attachment Facility (CAF)
- an implicit connect to DB2 UDB

**User Response:** See the DB2 codes in the messages and codes publication of the DB2 database manager on your operating system for the appropriate reason code.

For DB2 for z/OS errors, see the section in the administration guide that describes the Call Attachment Facility. Contact your DBA for questions and diagnosis.

If you are running the program under DB2 UDB for UNIX, ensure that the LIBPATH environment variable is set to the same environment in which the program starts. Refer to the Setting up for replication documentation in the DB2 Replication Guide and Reference for additional information.

---

**ASN0538I** *pgmname : program\_qualifier : The program is waiting for DB2 to come up.*

**Explanation:** When the program is initially brought up, if DB2 is not up at that time, the program waits until DB2 is up. After DB2 is up, the Capture program makes the connection and

begins to capture changes.

If the NOTERM option is specified in the Capture invocation parameters, and DB2 comes down smoothly, the Capture program waits for it to come back up.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0539E** *pgmname : program\_qualifier : Database or subsystem named database\_server\_name needs to be configured with LOGRETAIN=RECOVERY.*

**Explanation:** The Capture program tried to perform a cold or warm start and the source database was not defined properly in order for the log read interface to be used by the Capture program. The accepted settings for the database configuration parameter LOGRETAIN is RECOVERY (also known as ON).

**User Response:** Use the **update database configuration for** command to alter the setting of the LOGRETAIN parameter. Please note that when you set this parameter to RECOVERY (or ON), you must then use the **backup** command to backup the database before using this database with any application.

---

**ASN0540E** *pgmname : program\_qualifier : The program was not able to execute the autobind operation successfully on package pkg\_name from the file path\_filename. The SQLSTATE sqlstate was returned.*

**Explanation:** The program discovered that a bind or rebind is required in order to run. The program attempted to autobind, and the autobind was unsuccessful. The program failed to initialize.

**User Response:** Check for corresponding database messages that might provide additional details about the source of the autobind failure. Possible problems include authorization failures, missing or incorrect control tables, or bind files that do not match the program. Correct the situation, and restart the program.



---

**ASN0541E** *pgmname : program\_qualifier : An incorrect value column\_value was supplied for column column\_name of the program parameter table parms\_table.*

**Explanation:** This message is issued during initialization of the Capture program if the validation of the IBMSNAP\_CAPPARMS table found that one of the parameter value settings is not valid. The Capture program terminates with this error.

**User Response:** Check the documentation for permitted parameter values allowed in this table. Correct the values and restart the Capture program.

---

**ASN0542E** *pgmname : program\_qualifier : The maximum number of lock time-out or deadlock retries has been reached, SQLSTATE sqlstate was returned.*

**Explanation:** The Capture program has internally retried a time-out or deadlock condition multiple times. If the program task that receives the persistent lock condition is critical, such as the Capture worker thread, the whole Capture program terminates. If the program task is not critical, such as pruning or monitoring, then the task will be retried at a later time, and the Capture program remains active.

**User Response:** Check for corresponding database messages that might provide additional detail about the source of the locking contention. Correct the situation if the error is a user error, such as a user held lock. If the condition persists, contact IBM Software Support for assistance.

---

**ASN0543E** *pgmname : program\_qualifier : The program cannot obtain bytes\_number bytes of storage for a object.*

**Explanation:** The program is unable to obtain memory for a necessary in-memory storage structure. The program terminates.

**User Response:** Consider allowing a larger

memory allowance to the program, then restart the program.

---

**ASN0544E** *pgmname : program\_qualifier : The program is already active.*

**Explanation:** You tried to start more than one Capture program per DB2 subsystem or database.

- For VSE/ESA, Capture for VSE generates a unique lock name for each database. This lock name is already in use, indicating that Capture for VSE is already active for the database.
- For VSE/ESA, Capture for VM has determined that the resource ID used as a lock is already in use. The resource ID is specified on the ENQ\_NAME parameter of the CAPTURE ASNPARMs file.

**User Response:**

- For DB2 for z/OS subsystems, either run only one instance of the Capture program for all subsystems that are members of a data-sharing group, or run only one instance of the Capture program on any stand-alone system. Display the ENQ resource to determine the unique resource name violation.
- For other DB2 database operating systems, run only one Capture program per database.
- For Capture for VM, you can change the ENQ\_NAME parameter in the CAPTURE ASNPARMs to ensure unique values for each Capture program if you want to run Capture for VM for more than one DB2 database on a system.

---

**ASN0545E** *pgmname : program\_qualifier : The program started with the startup parameter PWDFILE, but the password file password\_file was not found.*

**Explanation:** The program cannot find the password file. The user specifies the password filename through the PWDFILE parameter. If the user specifies a path startup parameter, the password file should reside in the specified directory. If the user did not specify a path startup parameter, the password file should

reside in the current directory in which the program is running.

**User Response:** Ensure that the password file name is correctly specified and is located in the proper directory.

---

**ASN0546W** *pgmname : program\_qualifier : The program call issued to the Automatic Restart Manager failed. The invoked IXCARM macro is arm\_call, the return code is return\_code, and the reason code is reason\_code.*

**Explanation:** The Capture or Apply program cannot connect to, disconnect from, or receive a ready status indication from the Automatic Restart Manager (ARM). The message displays the unsuccessful call and the return or reason code that returned to the program from the ARM. The program does not terminate but cannot continue ARM processing.

**User Response:** Check the Automatic Restart Manager documentation for more information about the cause of this failure.

---

**ASN0547I** *pgmname : program\_qualifier : The number of substitution variables passed, nbr\_vars, does not match the number of tokens, nbr\_tokens, in the text of message number msg\_nbr.*

**Explanation:** The program code and the program message file do not match; the release level of the program and the message file catalog might not match.

**User Response:** Verify that the program message file is correctly installed with the appropriate file permission settings.

---

**ASN0548I** *pgmname : program\_qualifier : The program received an operator stop command.*

**Explanation:** This informational message indicates that a stop command was issued to the program.

**User Response:** This message is for your information only, and no action is required.

---

**ASN0552E** *pgmname : program\_qualifier : The program encountered an SQL error. The server name is server\_name. The SQL request is sql\_request. The table name is table\_name. The SQLCODE is sqlcode. The SQLSTATE is sqlstate. The SQLERRMC is sql\_tokens. The SQLERRP is error\_module.*

**Explanation:** A nonzero SQLCODE returned when the Capture, Apply, or Monitor program issued an EXEC SQL statement or CLI call. This SQLCODE might be caused by a DB2 problem that needs to be investigated, such as, an out of space condition, or DB2 is unavailable for use by applications. This message is sometimes followed by a second message that provides information about what replication was doing when this SQLCODE was encountered.

**User Response:** See the messages and codes documentation of the DB2 database manager on your operating system for an explanation of this SQLCODE and for information about corrective actions that might need to be taken in DB2. If replication issued another message immediately following this one, see the explanation and user response for that message.

---

**ASN0553E** *pgmname : program\_qualifier : Internal error error\_number occurred for message number msg\_number containing num\_tokens substitution fields: sub\_tokens.*

**Explanation:** The *error number* is a decimal internal error number which is defined as:

- |   |                       |
|---|-----------------------|
| 1 | Instance is not valid |
| 2 | Access denied         |
| 3 | No files              |
| 4 | No message            |
| 5 | Locale is not valid   |
| 6 | System error          |

The *msg\_number* is the message that the program was trying to issue. The *num\_tokens* is the number of substitution tokens given for the message (not including the *pgmname* and *program\_qualifier* tokens). The *sub\_tokens* is the substitution tokens for the message in error separated by commas.

**User Response:** Take any corrective action possible based on the error code given. For example, if the message file was not found or could not be accessed, you should also see message ASN0513 which gives you the file name. Verify that the message file exists with the correct permissions. If you get error code 4, you might have an old message file.

---

**ASN0554E** *pgmname : program\_qualifier : The program encountered a DB2 log full condition on server server\_name.*

**Explanation:** The program tried to process an insert or update which was denied by DB2 because the DB2 transaction log is full. The program will stop processing.

**User Response:** Check the amount of space remaining on the filesystem containing your database files. Consider increasing the maximum log size in the database configuration file.

---

**ASN0555W** *pgmname : program\_qualifier : The program cannot register with Automatic Resource Manager (ARM) because it is not APF authorized.*

**Explanation:** The Capture, Apply or Monitor program cannot register to use Automatic Resource Manager services because the program libraries are not APF authorized.

**User Response:** If you desire Capture, Apply, or Monitor program to register with Automatic Resource manager, authorize the program libraries for APF and restart the program.

---

**ASN0777I** *pgmname : program\_qualifier : Additional information message\_text, reason code(s): rc1, rc2, rc3.*

**Explanation:** The *Additional information* shown in this message refers to an informational text message. The reason codes provide supplemental return code information related to this message text. If an informational code field is not applicable, it contains "\*" (an asterisk).

**User Response:** This message is for your information only, and no action is required.

---

**ASN0888E** *pgmname : program\_qualifier : EEE error condition message\_text, error code(s): rc1, rc2, rc3.*

**Explanation:** The *EEE error condition* shown in this message is the description of an EEE-specific error that occurred in the specified program with the specified qualifier (if displayed). The error codes provide supplemental information related to this message text. If an error code field is not applicable, it contains "\*" (an asterisk).

**User Response:** Use the information from the *EEE error condition* and from the specified error codes to determine the cause of the error. Contact IBM Software Support if you cannot resolve the error.

---

**ASN0999E** *pgmname : program\_qualifier : Error condition message\_text, error code(s): rc1, rc2, rc3.*

**Explanation:** The *Error condition* shown in this message is the description of an error that occurred in the specified program with the specified qualifier (if displayed). The error codes provide supplemental information related to this message text. If an error code field is not applicable, it contains "\*" (an asterisk).

**User Response:** Use the information from the *Error condition* and from the specified error codes to determine the cause of the error. Contact IBM Software Support if you cannot resolve the error.

---

**ASN1001E** **APPLY** *apply\_qualifier*. The Apply program encountered an SQL error. The **ERRCODE** is *error\_code*. The **SQLSTATE** is *sqlstate*. The **SQLCODE** is *sqlcode*. The **SQLERRM** is *sqlerrm*. The **SQLERRP** is *sqlerrp*. The server name is *server\_name*. The table name is *table\_name*.

**Explanation:** An error occurred during the execution of an SQL statement.

**User Response:** Refer to your database message reference for an explanation of the SQL error code.

---

**ASN1002E** **APPLY** *apply\_qualifier*. The *table\_name* could not be locked. **ERRCODE** is *error\_code*, **SQLSTATE** is *sqlstate*, **SQLCODE** is *sqlcode*, **SQLERRM** is *sqlerrm*, **SQLERRP** is *sqlerrp*, server name is *server\_name*, table name is *table\_name*

**Explanation:** The Apply program could not lock the table.

**User Response:** Refer to your database message reference.

---

**ASN1003E** **APPLY** *apply\_qualifier*. The Apply program could not connect to the server *server*. The error code is *error\_code*. The **SQLSTATE** is *sqlstate*. The **SQLCODE** is *sqlcode*. The **SQLERRM** is *sqlerrm*. The **SQLERRP** is *sqlerrp*.

**Explanation:** The Apply program attempted to connect to the database and received a failing return code. There are many possible reasons why the Apply program could not connect to the database. For example, the Apply program would receive a failing return code if the database was down or too many users were accessing it.

**User Response:** Look up the **SQLCODE** in the DB2 messages and codes manual to determine

why the connection failed. Refer to the Setting up for replication documentation in the DB2 Replication Guide and Reference for information about storing user IDs and passwords.

Refer to your database message reference for an explanation of the SQL error code.

---

**ASN1006E** **APPLY** *apply\_qualifier*. The product registration module has unexpected content.

**Explanation:** The content of the registration module (ASNAPR61) for DB2 Replication is not as expected for this version of the DB2. No further use of the product is possible until you provide the correct registration module.

**User Response:** Verify that DB2 was installed without errors. If errors occurred, correct them and try again.

If DB2 installed without error and you are correctly accessing the feature-registration module (ASNAPR61), contact IBM Software Support for assistance.

---

**ASN1008E** **APPLY** *apply\_qualifier*. The subscription set with Apply qualifier *qualifier* and set name *set\_name* is not defined correctly. **ERRCODE** is *error\_code*.

**Explanation:** The subscription set is not defined correctly.

**User Response:** Make sure that the **WHOS\_ON\_FIRST** column in the subscription set table is specified correctly.

---

**ASN1009E** **APPLY** *apply\_qualifier*. There is no subscription set defined for Apply qualifier *qualifier*.

**Explanation:** There is no subscription set defined for Apply qualifier *qualifier*.

**User Response:** Define at least one subscription set for Apply qualifier *qualifier*.

---

**ASN1010E** *APPLY apply\_qualifier. The Apply program could not insert row row into the audit trail table due to the following error: error\_code.*

**Explanation:** This is an SQL return code indicating that the audit trail table was not set up with the same structure as the IBMSNAP\_APPLYTRAIL table.

**User Response:** Refer to the Table structures documentation in the DB2 Replication Guide and Reference and to your database SQL manual.

---

**ASN1011E** *APPLY apply\_qualifier. The copy request has incompatible source and target attributes. The SQL code is error\_code.*

**Explanation:** This is an SQL code indicating that the attributes of the target table must be compatible with the attributes of the source table.

**User Response:** Refer to the SOURCE\_STRUCTURE column in the register table for the compatibility of the source and target attributes.

---

**ASN1012E** *APPLY apply\_qualifier. The source table structure is not valid. The error code is error\_code.*

**Explanation:** This is an SQL return code indicating that the source table structure in the register table was not set up according to the SOURCE\_STRUCTURE column in the register table.

**User Response:** Refer to the Table structures documentation in the DB2 Replication Guide and Reference for valid SOURCE\_STRUCTURE column values used in the IBMSNAP\_REGISTER table.

---

**ASN1013E** *APPLY apply\_qualifier. The target table structure is not valid. The error code is error\_code.*

**Explanation:** The target table structure in the subscription-targets-member table was not valid.

**User Response:** Refer to the Table structures

documentation in the DB2 Replication Guide and Reference for valid target table structures.

---

**ASN1014E** *APPLY apply\_qualifier. The Apply program could not find a source for the copy request because it could not find the change data table. The error code is error\_code.*

**Explanation:** The change data table was not defined in the register table because either the Apply program did not find the change data table name in the register table or the source table was not registered correctly.

**User Response:** Refer to the Table structures documentation in the DB2 Replication Guide and Reference, and verify that the change data table is correctly defined in the register table.

---

**ASN1015I** *APPLY apply\_qualifier. The Apply program is waiting for the Capture program at server server\_name to advance the global SYNCHTIME. Verify that the Capture program is running.*

**Explanation:** This message is for your information only.

**User Response:** Verify that the Capture program is running.

---

**ASN1016I** *APPLY apply\_qualifier. Refresh copying has been disabled. The error code is error\_code.*

**Explanation:** While attempting to perform a full refresh, the Apply program encountered a DISABLE\_REFRESH column in the register table which was set on.

**User Response:** Either turn off the DISABLE\_REFRESH column or bypass the Apply program and perform a manual refresh.

---

**ASN1017E** **APPLY** *apply\_qualifier*. The Apply program could not find any target column names. The error code is *error\_code*.

**Explanation:** The Apply program could not find any columns in the subscription columns table.

**User Response:** Redefine the subscription set and subscription-set members. Refer to the Setting up for replication documentation in the DB2 Replication Guide and Reference for more information.

---

**ASN1018I** **APPLY** *apply\_qualifier*. The Apply program is processing subscription set *set\_name(whos\_on\_first).(set\_number of total\_sets)*.

**Explanation:** This message is for your information only.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1019E** **APPLY** *apply\_qualifier*. The target table does not have any key columns. The error code is *error\_code*.

**Explanation:** The Apply program cannot find key column names in one of the columns requiring a unique index or primary key.

**User Response:** Redefine the subscription set and the subscription-set members. Refer to the Setting up for replication documentation in the DB2 Replication Guide and Reference for more information.

---

**ASN1020E** **APPLY** *apply\_qualifier*. The Apply program could not reserve a storage block. The error code is *error\_code*.

**Explanation:** The Apply program could not obtain the required (memory) storage.

**User Response:** Contact IBM Software Support.

---

**ASN1021E** **APPLY** *apply\_qualifier*. The Apply program cannot read the work file *filename*. The error code is *error\_code*.

**Explanation:** The Apply program cannot read the work file due to a system error.

**User Response:** Determine if the problem is caused by lack of space, and contact your system administrator to obtain what is needed.

---

**ASN1022E** **APPLY** *apply\_qualifier*. The Apply program cannot write to the work file *filename*. The error code is *error\_code*.

**Explanation:** Either the user does not have the proper access authority for one or all of the files, or there is insufficient space remaining after writing to the target file.

**User Response:** Determine whether the problem is caused by a lack of access authority or a lack of space, and contact your system administrator to obtain what is needed.

---

**ASN1023E** **APPLY** *apply\_qualifier*. The Apply program cannot open the work file *filename*. The error code is *error\_code*.

**Explanation:** The Apply program cannot open the work file.

**User Response:** Contact IBM Software Support.

---

**ASN1024E** **APPLY** *apply\_qualifier*. The Apply program cannot close the work file *filename*. The error code is *error\_code*.

**Explanation:** The Apply program cannot close the work file.

**User Response:** Contact IBM Software Support.

---

**ASN1025I**    **APPLY** *apply\_qualifier*. The Apply program completed processing for subscription set *set\_name(whos\_on\_first)*. The return code is *return\_code*.

**Explanation:** This message is for your information only.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1026I**    **APPLY** *apply\_qualifier*. The Apply program encountered an error while trying to bind. **SQLSTATE** is *sqlstate*, **SQLCODE** is *sqlcode*.

**Explanation:** An error occurred during the execution of bind.

**User Response:** Refer to your database message reference.

---

**ASN1027E**    **APPLY** *apply\_qualifier*. There are too many large object (LOB) columns specified. The error code is *error\_code*.

**Explanation:** Too many large object (BLOB, CLOB, or DBCLOB) columns are specified for a subscription set member. The maximum number of columns allowed is 10.

**User Response:** Remove the excess large object columns from the subscription set member.

---

**ASN1028I**    **APPLY** *apply\_qualifier*. The before-image column for a key column is not found. The error code is *error\_code*.

**Explanation:** The subscription set up for a member with **TARGET\_KEY\_CHG=Y** is incorrect.

**User Response:** For each key column (**IS\_KEY=Y**), there must be a before-image column included in the **IBMSNAP\_SUBS\_COLS** table. It can be a **col\_type=B** (specified by the user), or **col\_type=P** (provided by Replication). If the subscription is set up manually, then you must correct the problem yourself. If the

subscription is set up by using the Replication Center or the Replication Commands, contact IBM Software Support.

---

**ASN1029E**    **APPLY** *apply\_qualifier*. The SQL statement of the subscription set named *set\_name* with a *whos\_on\_first* value of *whos\_on\_first* did not execute successfully. The statement failed with **SQLCODE** *sqlcode* and **SQLSTATE** *sqlstate*. The apply program internal error code is *error\_code*.

**Explanation:** The user-specified SQL statement did not execute successfully.

**User Response:** Refer to the corresponding information in the **IBMSNAP\_APPLYTRAIL** table and to the SQL manual of your database for detailed information.

---

**ASN1031E**    **APPLY** *apply\_qualifier*. The SQL statement is empty. The error code is *error\_code*.

**Explanation:** The SQL statement is an empty string.

**User Response:** Specify the SQL statement to be executed.

---

**ASN1032E**    **APPLY** *apply\_qualifier*. The Apply program log file could not be opened. The error code is *error\_code*, and the return code is *return\_code*.

**Explanation:** The Apply program could not open the log file.

**User Response:** For more information on the return code, refer to the manual that describes problem determination for your particular operating system.

---

**ASN1033E** *APPLY apply\_qualifier. The Apply program could not write to the Apply log file. The error code is error\_code, and the return code is return\_code.*

**Explanation:** The Apply program could not write to the log file.

**User Response:** For more information on the return code, refer to the manual that describes problem determination for your particular operating system.

---

**ASN1034I** *APPLY apply\_qualifier. The Apply program initialization is successful.*

**Explanation:** This message is issued at successful initialization of the Apply process.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1035E** *APPLY apply\_qualifier. The Apply program could not access the subscription columns table. The error code is error\_code. The SQLSTATE is sqlstate. The SQLCODE is sqlcode. The SQLERRM is sqlerrm. The SQLERRP is sqlerrp. The server name is server\_name. The table name is table\_name.*

**Explanation:** An error occurred during the execution of an SQL statement.

**User Response:** Refer to your database message reference for SQL.

---

**ASN1036E** *APPLY apply\_qualifier. The column type col\_type for expression expression is not valid. The error code is error\_code.*

**Explanation:** The value for the COL\_TYPE column in the subscription columns table is not valid.

**User Response:** Change the value to A, B, C, D, F, L, or R.

---

**ASN1038E** *APPLY apply\_qualifier. No column names or expressions were specified in the subscription columns table.*

**Explanation:** Column names or expressions for a copy statement must be specified.

**User Response:** Refer to the Setting up for replication documentation in the DB2 Replication Guide and Reference for more information about requirements for subscription definitions.

---

**ASN1039E** *APPLY apply\_qualifier. The Apply program plan, plan\_name, could not be opened. The error code is error\_code. The return code is return\_code. The reason code is reason\_code.*

**Explanation:** The Apply program plan could not be opened.

**User Response:** Refer to the Apply for z/OS Program Directory.

---

**ASN1040E** *APPLY apply\_qualifier. The Apply program encountered an z/OS error. The error code is error\_code, and the return code is return\_code.*

**Explanation:** Execution of a z/OS system operation failed.

**User Response:** Refer to your z/OS system library information.

---

**ASN1041I** *APPLY apply\_qualifier. The Apply program was started using subsystem name: subsystem.*

**Explanation:** This message informs you that the Apply program started using the specified subsystem name.

**User Response:** This message is for your information only, and no action is required.



---

**ASN1042W** *APPLY apply\_qualifier. There are too many invocation parameters.*

**Explanation:** The number of parameters you specified when you invoked the Apply program exceeds the maximum allowed.

**User Response:** Refer to the Capture and Apply chapter for your operating system for information on the appropriate number of invocation parameters.

---

**ASN1043E** *APPLY apply\_qualifier. There is already one Apply instance running with this Apply program qualifier qualifier. The error code is error\_code, and the reason code is reason\_code.*

**Explanation:** Verification attempt failed.

**User Response:** Make sure that only one instance of the Apply program with the specified Apply qualifier is running under this user ID on this subsystem or database.

---

**ASN1044I** *APPLY apply\_qualifier. The Apply program will become inactive for number minutes and number seconds.*

**Explanation:** The Apply program is inactive.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1045I** *APPLY apply\_qualifier. The Apply program was started using database database.*

**Explanation:** This message informs you from which database the Apply program is running.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1047I** *APPLY apply\_qualifier. There are too many columns specified. The error code is error\_code.*

**Explanation:** There are too many columns specified for a member in the subscription.

**User Response:** The user must reduce the number of columns specified for the member in the subscription. The maximum number of columns supported is determined by the total length of all the column names. More columns can be specified if the length of the column names is shorter.

---

**ASN1048E** *APPLY apply\_qualifier. The execution of an Apply cycle failed. See the Apply trail table for full details: text*

**Explanation:** An Apply cycle failed. In the message, *text* identifies the *target\_server*, *target\_owner*, *target\_table*, *stmt\_number*, and *cntl\_server*.

**User Response:** Check the APPERRM fields in the audit trail table to determine why the Apply cycle failed.

---

**ASN1049E** *APPLY apply\_qualifier. The Apply program encountered a system error. The error code is error\_code. The return code is return\_code.*

**Explanation:** Execution of a system operation failed.

**User Response:** Refer to the system library information for your operating system.

---

**ASN1050E** *APPLY apply\_qualifier. The Apply program encountered an operation that is not valid while updating the target table. The error code is error\_code. The operation to be applied is operation.*

**Explanation:** The operation field of a row fetched from the source table is not valid.

**User Response:** Contact IBM Software Support.

---

**ASN1051E** *APPLY apply\_qualifier. The Apply program detected a gap between the source source table and the target table. The error code is error\_code.*

**Explanation:** The Apply program has detected that the Capture program lost change data before the Apply program could copy it. For example, the Capture program might have been cold started or retention limit pruning might have occurred.

**User Response:** Check the control tables to determine why the gap is present. Take proper action to preserve data integrity before you reset the control table information to execute the definition again.

---

**ASN1052E** *APPLY apply\_qualifier. The Apply program could not find the ASNLOAD program.*

**Explanation:** The Apply program cannot find the ASNLOAD program in the current directory.

**User Response:** Make sure that ASNLOAD is in the directory from which you are invoking the Apply program.

---

**ASN1053E** *APPLY apply\_qualifier. The execution of the ASNLOAD exit routine failed. The return code is return\_code.*

**Explanation:** The ASNLOAD exit routine detected an error and passed the error information back to the Apply program. The following values are valid return codes:

98

unexpected error has occurred (The ASNLOAD exit routine has failed with an unexpected error. No processing will be performed.)

99

DB2 pwdfile keyword supplied - password file not found (The pwdfile parameter was passed, but no password file was found. This is an error, and no connections or other processing will be performed.)

100

DB2 connect with a user/using phrase failed - pwdfile found (A connect with a user/using phrase was made using values supplied in the encrypted Apply password file. The password file and a userid/password combination for the DB2 server were found, but the connection failed.)

101

DB2 connect without a user/using phrase failed - no pwdfile found (A connection without a user/using phrase was made because no password file was provided. The connection failed.)

102

DB2 connect without a user/using phrase failed - pwdfile found, no entry (A connection without a user/using phrase was made, because no server entry was found in the pwdfile for the DB2 server. The connection failed.)

103

DB2 connect with a user/using phrase failed - uid/pwd from asnload.ini used (A connection with a user/using phrase was made using values supplied in the asnload.ini file. This file and a userid/password combination for the DB2 server were found, but the connection failed.)

104

DB2 connect without a user/using phrase failed - no asnload.ini found (A connection without a user/using phrase was made, because no asnload.ini file was found. The connection failed.)

105

DB2 connect without a user/using phrase failed - no uid/pwd found for server (A connection without a user/using phrase was made. The asnload.ini file was found, but no uid/pwd combination was provided. The connection failed.)

106

user specified LOADX\_TYPE = 2, no user code provided (The value of LOADX\_TYPE in the

table ASN.IBMSNAP\_SUBS\_MEMBR was set by the user to the value of 2, indicating that the user was supplying custom code in the ASNLOAD exit routine. However this code was not found, and the ASNLOAD exit routine failed when the Apply program passed a LOADX\_TYPE value of 2.)

107

DB2 import utility failed (The import utility failed to execute. The SQL code returned by the utility is passed as the reason code.)

108

DB2 export utility failed (The export utility failed to execute. The SQL code returned by the utility is passed as the reason code.)

109

DB2 load utility failed (The load utility failed to execute. The SQL code returned by the utility is passed as the reason code.)

110

DB2 load utility failed - invoked as crossload (The load utility failed to execute. The load utility was invoked with the load from cursor option. The SQL code returned by the utility is passed as the reason code.)

111

user has set LOADX\_TYPE to an not valid value (The ASNLOAD exit routine was invoked with a LOADX\_TYPE value that was set by the user. The LOADX\_TYPE value is not valid for this environment, and the ASNLOAD exit routine failed.)

112

LOADX\_TYPE 3 requires a nickname for select (The ASNLOAD exit routine failed. The ASNLOAD exit routine was invoked with a LOADX\_TYPE value that was set by the user. The LOADX\_TYPE value is not valid for this environment unless a nickname is created for the remote DB2 source table and stored in the ASN.IBMSNAP\_SUBS\_MEMBR table.)

113

LOADX\_TYPE 4 is incompatible with target table

(The ASNLOAD exit routine failed. The ASNLOAD exit routine was invoked with a LOADX\_TYPE set by the user. The LOADX\_TYPE value is not valid for this environment, because the target table cannot be serviced by the DB2 Linux/UNIX/Windows load utility.)

114

LOADX\_TYPE 5 is incompatible with target table (The ASNLOAD exit routine failed . The ASNLOAD exit routine was invoked with a LOADX\_TYPE set by the user. The LOADX\_TYPE value is not valid for this environment, because the target table cannot be serviced by the DB2 import utility.)

115

the ASNDLCOPY exit routine has failed (The ASNLOAD exit routine called the ASNDLCOPY exit routine, because there were DATALINK columns for the subscription-set member. The ASNDLCOPY exit routine failed; therefore, the process that loads this subscription-set member also failed.)

**User Response:** Check the return code and the corresponding explanation (above). Check for additional information in the ASNLOAD message file and in the message files generated by the DB2 utility, if applicable.

---

**ASN1054E** *APPLY* *apply\_qualifier*. The Apply program could not find a row in the IBMSNAP\_REGISTER or IBMSNAP\_PRUNCNTL table that corresponds to the subscription set member with a set name *set\_name*, for source owner *src\_owner*, source table *src\_tbl*, and source view qualifier *src\_view\_qual*.

**Explanation:** The source table registration is incorrect or incomplete.

**User Response:** Drop and redefine the registration.

---

**ASN1055E** **APPLY** *apply\_qualifier*. The Apply program could not find the prune control information for source owner *src\_ownr*, source table *src\_tbl*, source view qualifier *src\_view\_qual*, target owner *tgt\_ownr*, and target table *tgt\_tbl*.

**Explanation:** The source table registration is incorrect.

**User Response:** Drop the subscription and redo it.

---

**ASN1056E** **APPLY** *apply\_qualifier*. The Apply program could not connect to the server due to lack of user ID/password. The error code is *error\_code*.

**Explanation:** The Apply program could not find the password and user ID to connect to the server.

**User Response:** Make sure that the Apply program password file exists. The Apply program password file resides in the same directory from which you start the Apply program. If you are using DB2 Universal Database Satellite Edition, make sure that the password and user ID are defined to the client systems.

---

**ASN1057E** **APPLY** *apply\_qualifier*. The Apply program could not read the password in the Apply password file. The error code is *error\_code*.

**Explanation:** The Apply program found no password.

**User Response:** If you want to use the AUTHENTICATION=SERVER scheme, you must provide a password, as described in the Apply program section in the Capture and Apply chapter for your operating system.

---

**ASN1058E** **APPLY** *apply\_qualifier*. The Apply program could not close the password file. The error code is *error\_code*.

**Explanation:** The Apply program could not close the password file.

**User Response:** Contact IBM Software Support.

---

**ASN1059E** **APPLY** *apply\_qualifier*. The Apply program detects syntax that is not valid for line *line* in the password file. The error code is *error\_code*.

**Explanation:** The Apply program could not recognize a line in the password file.

**User Response:** Correct the syntax error in the password file. See the Apply program section in the Capture and Apply chapter for your operating system.

---

**ASN1060E** **APPLY** *apply\_qualifier*. The dynamic allocation for the temporary work file failed. The error code is *error\_code*.

**Explanation:** A system error was encountered during dynamic allocation.

**User Response:** Contact IBM Software Support.

---

**ASN1061E** **APPLY** *apply\_qualifier*. The specified keyword parameter is not valid. The error code is *error\_code*.

**Explanation:** An invocation parameter that is not valid has been specified and has been ignored by the Apply program.

**User Response:** Correct the invocation parameter. See the Apply program section in the Capture and Apply chapter for your operating system.

---

**ASN1062W** *APPLY apply\_qualifier. The Apply program must use SELECT and INSERT statements to perform a full refresh of this subscription-set member. The following information pertains to this subscription-set member: the set name is set\_name, the source owner is source\_owner, the source table is source\_table, the source view qualifier is source\_view\_qual, the target owner is target\_owner, and the target table is target\_table.*

**Explanation:** The ASNLOAD exit routine cannot detect a user-specified LOADX\_TYPE value, and no utilities are available to process this subscription-set member. Therefore, the ASNLOAD exit routine passes full refresh control back to the Apply program. The ASNLOAD exit routine does not currently support and is not able to process some target table types (such as the Sybase and MS SQL Server target tables).

**User Response:** This message is for your information only, and no action is required. However, you can set the value of the LOADX\_TYPE to 1 for these subscription-set members in order to avoid unnecessary processing by the ASNLOAD exit routine.

---

**ASN1063E** *APPLY apply\_qualifier. A subscription set cannot have more than 200 members. The error code is error\_code.*

**Explanation:** The number of subscriptions has exceeded the maximum allowed number of 200.

**User Response:** Remove excess members from the subscription set.

---

**ASN1064W** *APPLY apply\_qualifier. The Apply program cannot perform a full refresh for the subscription set named set\_name, because the Capture program for this source has not yet been cold started.*

**Explanation:** The Apply program cannot attempt a full refresh for the subscription set,

because the Capture program for this source has never been cold started and is not ready to process the CAPSTART signals are be inserted by the Apply program.

**User Response:** Start the Capture program for this source.

---

**ASN1065E** *APPLY apply\_qualifier. The Apply program cannot perform a full refresh for the subscription set named set\_name, because the registered source table source\_owner.source\_table is in a stopped state.*

**Explanation:** The Apply program cannot perform a full refresh for this subscription set, because one of the source registrations is in the stopped state (the value of the STATE column is set to 'S' in the IBMSNAP\_REGISTER table). The Capture program places a registration in the stopped state if there is a problem with the registration that requires user intervention. The integrity of the captured data for the registration might be compromised, and the Apply program must perform a full refresh. (This could occur if the registered source table has been altered with data capture none.)

**User Response:** Fix the stopped registrations using the information from the generated Capture error messages. Manually set the value of the STATE column to 'I' (Inactive) in the IBMSNAP\_REGISTER table. The Apply program can then perform a full refresh.

---

**ASN1066E** *APPLY apply\_qualifier. An internal Apply program error occurred. The error code is error\_code.*

**Explanation:** An internal Apply program error occurred.

**User Response:** Contact IBM Software Support.

---

**ASN1067E** **APPLY** *apply\_qualifier*. The Apply program has detected update conflicts and compensated rejected transactions. See the unit-of-work table for details. The error code is *error\_code*.

**Explanation:** More than one application updated the same row in a table from different locations. Some transactions have been rejected and compensated.

**User Response:** Refer to the Tables structures documentation in the DB2 Replication Guide and Reference for more information.

---

**ASN1068E** **APPLY** *apply\_qualifier*. The Apply program has deactivated the subscription due to an RI violation. The error code is *error\_code*.

**Explanation:** A referential integrity violation was detected when copying data from the source table to a replica. The Apply program has terminated and the subscription has been deactivated.

**User Response:** Correct the referential integrity error and reactivate the subscription.

---

**ASN1070E** **APPLY** *apply\_qualifier*. The Apply program could not lock the target table. The *ERRCODE* is *error\_code*. The *SQLSTATE* is *sqlstate*. The *SQLCODE* is *sqlcode*. The *SQLERRM* is *sqlerrm*. The *SQLERRP* is *sqlerrp*. The server name is *server\_name*. The table name is *table\_name*.

**Explanation:** The Apply program could not lock the target tables before it was to check update conflicts.

**User Response:** Verify that all the target tables are available before restarting Apply.

---

**ASN1071E** **APPLY** *apply\_qualifier*. The Apply program could not reposition the work file. The error code is *error\_code*.

**Explanation:** The Apply program has detected an error while reading the temporary work file.

**User Response:** Contact IBM Software Support.

---

**ASN1072E** **APPLY** *apply\_qualifier*. The Apply program could not find the *ASNDONE* program.

**Explanation:** The Apply program could not find the user exit program, *ASNDONE*.

**User Response:** Verify that the *ASNDONE* program is located in the correct directory.

---

**ASN1073E** **APPLY** *apply\_qualifier*. The execution of the *ASNDONE* program failed. The return code is *return\_code*.

**Explanation:** An error occurred while calling the user exit program, *ASNDONE*.

**User Response:** Contact IBM Software Support.

---

**ASN1074E** **APPLY** *apply\_qualifier*. The Apply program could not find the *ASNDLCOPY* program.

**Explanation:** The Apply program did not find the *ASNDLCOPY* program in the current search path.

**User Response:** Add the *ASNDLCOPY* program to the search path and run the Apply program again.

---

**ASN1075E** **APPLY** *apply\_qualifier*. The *ASNDLCOPY* program failed. The return code is *return\_code*. Additional information can be found in the *ASNDL appl\_qualset\_namecapture\_server target\_server.LOG* file.

**Explanation:** The *ASNDLCOPY* program detected an error and passed the error

information back to the Apply program. The following values are valid return codes:

98

Unexpected error occurred.

99

The arguments passed to the ASNDLCOPY program are not valid.

100

Unable to allocate memory.

101

Unable to open the ASNDLSRVMAP configuration file.

102

The number of entries in the ASNDLSRVMAP configuration file exceeds the maximum limit.

103

An entry that is not valid has been found in the ASNDLSRVMAP configuration file.

104

No user login information was found in the ASNDLUSER configuration file for a given file server.

105

An entry that is not valid has been found in the ASNDLPARM configuration file.

106

Unable to open the ASNDLUSER configuration file.

107

An entry that is not valid has been found in the ASNDLUSER configuration file.

108

An I/O error occurred when reading from the input file.

109

An entry that is not valid has been found in the input file.

110

Unable to open the input file.

111

Unable to open the result file.

112

An I/O error occurred when writing to the result file.

113

An error occurred when initializing the control channel of the FTP protocol.

114

An error occurred when sending data through the control channel.

115

Unable to log on to the file server with the given user and password.

116

The copy daemon has not yet started.

117

An error occurred when initializing the data channel of the FTP protocol.

118

Unable to retrieve the file from the source file server.

119

Unable to store the file on the target file server.

120

An error occurred when transferring files in the passive mode.

121

Cannot find the path mapping for the given file reference.

122

An error occurred when executing the FTP BINARY command.

123

An error occurred when executing the FTP SIZE command.

124

An error occurred when executing the FTP MODTIME command.

125

An error occurred when executing the FTP SITE UMASK command.

126

An error occurred when executing the FTP SITE TOUCH command.

127

An error occurred when executing the FTP SITE CHMOD command.

**User Response:** Check the return code and its corresponding meaning (above). The return code is based on the sample ASNDLCOPY program that is shipped with the product. Additional information is provided in the log file.

---

**ASN1076E    The Apply program cannot read the format of the result file that was generated by the ASNDLCOPY program.**

**Explanation:** The result file that was generated by the ASNDLCOPY program is not in an expected format.

**User Response:** If you modified the ASNDLCOPY program, check that your changes are not causing the invalid format. If your changes are not the cause of the problem, check that your machine has enough space for the result file.

---

**ASN1077E    APPLY *apply\_qualifier*. The Apply program encountered an DATALINK column value that is not valid while updating the target table. The error code is *error\_code*.**

**Explanation:** The DATALINK column field of a row fetched from the source table is not valid.

**User Response:** Contact IBM Software Support.

---

**ASN1078E    APPLY *apply\_qualifier*. The ASNDLCOPY program was terminated by the signal *signal\_number*. Additional information can be found in the *filename* file.**

**Explanation:** The ASNDLCOPY program terminated abnormally by the given signal.

**User Response:** Check the specified log file for the cause of the error. If you modified the ASNDLCOPY program and the signal is generated by the modified code, fix the code and rerun. Otherwise, contact IBM Software Support.

---

**ASN1097I    APPLY *apply\_qualifier*. The Apply program stopped.**

**Explanation:** The error reported previously caused the Apply program to stop.

**User Response:** Fix the error reported before this message.

---

**ASN1207E    APPLY *apply\_qualifier*. The subscription for *subscription* was not activated.**

**Explanation:** The selected subscription is inactive.

**User Response:** Either activate the subscription or select another one.

---

**ASN1210E    APPLY *apply\_qualifier*. An Apply qualifier must be specified following the keyword -q.**

**Explanation:** You must specify an Apply qualifier following the keyword -q.

**User Response:** Specify an Apply qualifier following the keyword -q.



---

**ASN1212E** *APPLY apply\_qualifier. A read-only set name set\_name is found following the keyword keyword.*

**Explanation:** A read-only set name was specified following the keyword U or D.

**User Response:** Specify only replica for the keywords U and D.

---

**ASN1221I** *APPLY apply\_qualifier. Set set\_name has been successfully refreshed with number rows at time.*

**Explanation:** This message is for your information only.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1242E** *APPLY apply\_qualifier. An SQL error occurred. ERRCODE is error\_code, SQLSTATE is sqlstate, SQLCODE is sqlcode, SQLERRM is sqlerrm, SQLERRP is sqlerrp, table name is table\_name.*

**Explanation:** This message is for your information only.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1243E** *APPLY apply\_qualifier. There is no eligible subscription in the ASN.IBMSNAP\_SUBS\_SET table.*

**Explanation:** Either a subscription set has not been selected or the apply qualifier is not valid.

**User Response:** Verify the subscription names and apply qualifier.

---

**ASN1304E** *APPLY apply\_qualifier. The ASNSAT program terminated due to a Capture error.*

**Explanation:** The Capture program returned an error.

**User Response:** Determine the error from the Capture log file.

---

**ASN1305E** *APPLY apply\_qualifier. The ASNSAT program terminated due to an Apply error.*

**Explanation:** The Apply program returned an error.

**User Response:** Determine the error from the Apply log file.

---

**ASN1310E** *APPLY apply\_qualifier. The ASNSAT program encountered a system error while attempting to invoke the Capture program. Return code is return\_code.*

**Explanation:** An operating system error occurred while calling ASNCAP.

**User Response:** Make sure that the Capture program is in the execution path.

---

**ASN1311E** *APPLY apply\_qualifier. The ASNSAT program encountered a system error while attempting to invoke the Apply program. Return code is return\_code.*

**Explanation:** An operating system error occurred while calling ASNAPPLY.

**User Response:** Make sure that the Apply program is in the execution path.

---

**ASN1312E** *APPLY apply\_qualifier. The default target server, DB2DBDFT, is not set.*

**Explanation:** The user did not specify the target server name, and the ASNSAT program could not determine the default database name from DB2DBDFT.

**User Response:** Specify the target server name following the -t keyword.

---

**ASN1314E** **APPLY** *apply\_qualifier*. An SQL error occurred while ASNSAT was getting the default Apply qualifier. SQLSTATE is *sqlstate*, SQLCODE is *sqlcode*.

**Explanation:** The user did not specify the Apply qualifier. The ASNSAT program encountered an error while retrieving the USER special register.

**User Response:** Specify the Apply qualifier following the -q keyword.

---

**ASN1315E** **APPLY** *apply\_qualifier*. Cannot connect to database server. SQLSTATE is *sqlstate*, SQLCODE is *sqlcode*.

**Explanation:** An error occurred while attempting to connect to the target database.

**User Response:** Refer to your database message reference.

---

**ASN1316E** **APPLY** *apply\_qualifier*. ASNSAT encountered an error while trying to bind. The SQLSTATE is *sqlstate*, SQLCODE is *sqlcode*.

**Explanation:** An error occurred while attempting to auto bind.

**User Response:** Make sure that the bind file exists in the sqllib\bnd directory.

---

**ASN1317E** **APPLY** *apply\_qualifier*. An SQL error occurred while ASNSAT was getting the CD\_TABLE value from ASN.IBMSNAP\_REGISTER table. SQLSTATE is *sqlstate*, SQLCODE is *sqlcode*.

**Explanation:** An SQL error occurred while selecting from the register table.

**User Response:** Refer to your database message reference.

---

**ASN1318E** **APPLY** *apply\_qualifier*. An SQL error occurred while ASNSAT attempted to get the DB2 node type. SQLSTATE is *sqlstate*, SQLCODE is *sqlcode*.

**Explanation:** An error occurred while retrieving the node type configuration parameter.

**User Response:** Refer to your database message reference.

---

**ASN1500I** The Replication action *action\_name* started at timestamp with architecture level *architecture\_level*. The Capture server is *capture\_serveralias* and the Capture schema is *capture\_schema*.

**Explanation:** Valid values for *action name* are *Create Capture server control tables* and *Drop Capture server control tables*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1501I** The Replication action *action\_name* started at timestamp with architecture level *architecture\_level*. The Capture server is *capture\_serveralias*, the remote server is *remote\_servername*, and the Capture schema is *capture\_schema*.

**Explanation:** Valid values for *action name* are *Create Capture server control tables* and *Drop Capture server control tables*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1502I** The Replication action *action\_name* started at timestamp with architecture level *architecture\_level*. The Apply control server is *apply\_serveralias*.

**Explanation:** Valid values for *action name* are *Create Apply server control tables* and *Drop Apply server control tables*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1503I**    **The Replication action *action\_name* started at *timestamp*. The Capture server is *capture\_serveralias*, the Capture schema is *capture\_schema*, the source owner is *source\_owner*, and the source table, view, or nickname is *source\_table*.**

**Explanation:** Valid values for *action name* are *Create Registration*, *Drop Registration*, *Alter Registration*, *Add Registration*, and *Promote Registration*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1504I**    **The Replication action *action\_name* started at *timestamp*. The Capture server is *capture\_serveralias*, the remote server is *remote\_server*, the Capture schema is *capture\_schema*, the source owner is *source\_owner*, and the source table, view, or nickname is *source\_table*.**

**Explanation:** Valid values for *action name* are *Create Registration* and *Drop Registration*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1505I**    **The Replication action *action\_name* started. The subscription set information follows: the Apply control server is *control\_server*, the Apply qualifier is *apply\_qualifier*, the set name is *set\_name*, the target server is *target\_server* for remote server *remote\_servername*, the Capture server is *capture\_server* for remote server *remote\_servername*, and the Capture schema is *capture\_schema*.**

**Explanation:** Valid values for *action name* are *Create Subscription Set*, *Drop Subscription Set*, *Alter Subscription Set*, and *Promote Subscription Set*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1506I**    **The Replication action *action\_name* started at *timestamp*. The subscription set information follows: the Apply control server is *control\_server*, the Apply qualifier is *apply\_qualifier*, the set name is *set\_name*, the target server is *target\_server*, the Capture server is *capture\_server* for remote server *remote\_servername*, and the Capture schema is *capture\_schema*.**

**Explanation:** Valid values for *action name* are *Create Subscription Set*, *Drop Subscription Set*, *Alter Subscription Set*, and *Promote Subscription Set*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1507I**    **The Replication action *action\_name* started at *timestamp*. The subscription set information follows: the Apply control server is *control\_server*, the Apply qualifier is *apply\_qualifier*, the set name is *set\_name*, the target server is *target\_server* for remote server *remote\_server*, the Capture server is *capture\_server*, and the Capture schema is *capture\_schema*.**

**Explanation:** Valid values for *action name* are *Create Subscription Set*, *Drop Subscription Set*, *Alter Subscription Set*, and *Promote Subscription Set*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1508I**    **The Replication action *action\_name* started at *timestamp*. The subscription set information follows: the Apply control server is *control\_server*, the Apply qualifier is *apply\_qualifier*, the set name is *set\_name*, the target server is *target\_server*, the Capture server is *capture\_server*, and the Capture schema is *capture\_schema*.**

**Explanation:** Valid values for *action name* are *Create Subscription Set*, *Drop Subscription Set*, *Alter Subscription Set*, and *Promote Subscription Set*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1510I**    **The Replication action *action\_name* ended successfully at *timestamp*.**

**Explanation:** Valid values for *action name* are *Create Capture server control tables*, *Drop Capture server control tables*, *Create Apply control server control tables*, and *Drop Apply control server control tables*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1511I**    **The Replication action *action\_name* ended successfully for source owner *source\_owner* and source table, view, or nickname *source\_table*.**

**Explanation:** Valid values for *action name* are *Create Registration*, *Drop Registration*, *Alter Registration*, *Add Registration Column*, and *Promote Registration*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1512I**    **The Replication action *action\_name* ended successfully for Apply qualifier *apply\_qual*, set name *set\_name*.**

**Explanation:** Valid values for *action name* are *Create Subscription Set*, *Drop Subscription Set*, *Alter*

*Subscription Set*, *Add Statements to Subscription Set*, *Drop Statements from Subscription Set*, and *Promote Subscription Set*.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1513I**    **The Replication action *action\_name* ended successfully for Apply qualifier *apply\_qual*, set name *set\_name*, who's on first *whos\_on\_first*, source owner *source\_owner*, source table *source\_table*, source view qualifier *source\_view\_qual*, target owner *target\_owner*, and target table *target\_table*.**

**Explanation:** The following values are valid for *action name*:

- *Add Subscription Member*
- *Add Subscription Member Column*
- *Drop Subscription Member*

**User Response:** This message is for your information only, and no action is required.

---

**ASN1514I**    **The Replication action ended at *timestamp* with successful successes, errors errors, and warning warnings.**

**Explanation:** This message is for your information only.

**User Response:** No action is required.

---

**ASN1550E**    **The Replication action *action\_name* ended in error. The value for the input parameter *input\_parameter* is missing.**

**Explanation:** The input parameter is mandatory for this action and is missing.

**User Response:** Provide the mandatory parameter and rerun the Replication action.

**ASN1551E The Replication action *action\_name* ended in error. The value *value* for the input parameter *input\_parameter* is incorrect. The reason code is *reason\_code*.**

**Explanation:** The value provided for the input parameter is not a valid value. The following values are valid for the *reason\_code*:

- 0** Blocking minutes value should be between 0-999.
- 1** Commit Count value should be between 0-999.
- 2** Server Type value should be Capture Server.
- 3** Table type value should be one of the following types:
  - USERTABLE
  - CCD TABLE
  - POINT IN TIME
  - BASE AGGREGATE
  - CHANGE AGGREGATE
  - REPLICA
  - USERCOPY
- 4** Remote Server Name value should be NULL.
- 5** Server Type value should be one of the following types:
  - Capture Server
  - Control Server
  - Capture and Control Server
  - Capture, Control and Target Server
- 6** Internal CCD tables must be noncomplete.
- 7** The Apply qualifier exceeds the maximum length of 18 characters.
- 8** The set name exceeds the maximum length of 18 characters.
- 9** Event names must be 18 characters or fewer in length.

- 10** The source Capture schema name exceeds the maximum length of 30 characters.
- 11** The target Capture schema name exceeds the maximum length of 30 characters.
- 12** The BEFORE\_OR\_AFTER statement value must be 'A', 'B', or 'S'.
- 13** The EI\_OR\_CALL value must be 'C' or 'E'.
- 14** SQLSTATES must be 50 digits or fewer in length.
- 15** SQLSTATES must be numeric
- 16** The CONFLICT\_LEVEL must be zero (0) or NONE.
- 17** The CHGONLY value must be 'N'.
- 18** The external CCD table is noncondensed and contains LOB columns.
- 19** The CONFLICT\_LEVEL must be between 0 and 2.
- 20** The CHGONLY value must be 'Y' or 'N'.
- 21** The RECAPTURE value must be 'Y' or 'N'.
- 22** The DISABLE\_REFRESH value must be 0 or 1.
- 23** The CHG\_UPD\_TO\_DEL\_INS value must be 'Y' or 'N'.
- 24** The STOP\_ON\_ERROR value must be 'Y' or 'N'.
- 25** The BEFORE\_IMG\_PREFIX value must be only one character.
- 26** The corresponding tablespace does not have the *New Tablespace* flag set to true in any of the previous scenarios.
- 27** The table name is not a valid control table. Refer to the Replication Guide and Reference for a valid list of control tables.

**User Response:** Provide a valid value for the

input parameter, and rerun the replication action. See the online help for details.

---

**ASN1552E    The Replication action *action\_name* ended in error. The value *value1* for input parameter *input\_parameter1* is incompatible with the value *value2* for input parameter *input\_parameter2*.**

**Explanation:** The value provided for the Replication parameter clashes with another parameter specification.

**User Response:** Provide valid values for the input parameters and rerun the Replication action. See the online help for details.

---

**ASN1553E    The value *value1* for input parameter *input\_parameter1* is incompatible with the value *value2* for the existing subscription set *subscription\_set*, Apply qualifier *apply\_qual*, and who's on first *whoson\_first*.**

**Explanation:** The value provided for the Replication parameter clashes with one of the values for the existing subscription set.

**User Response:** Provide a valid value for the input parameter or change the subscription set definition, and rerun the Replication action. See the online help for details.

---

**ASN1560E    The replication action ended in error. An SQL error was encountered. SQL Message: *sql\_message*.**

**Explanation:** An error occurred during the execution of an SQL statement.

**User Response:** Refer to your database message reference for SQL.

---

**ASN1561E    Connection to the server *server\_alias* cannot be established. An SQL error was encountered. SQL message: *sql\_message*.**

**Explanation:** The connection to the specified server could not be established.

**User Response:** Refer to your database message reference for SQL. Verify that the userid and password information is correct.

---

**ASN1562E    The Replication action ended in error. An unexpected error occurred. Reference Code *reference\_code*.**

**Explanation:** The specified action cannot be performed because of a run time error.

**User Response:** Contact IBM Software Support.

---

**ASN1563E    The Replication action *action\_name* ended in error. The Replication architecture level *arch\_level* does not support server *server\_alias*.**

**Explanation:** The specified Replication architecture level is not supported on the specified server operating system, version, or release. The only valid value for *arch\_level* at this time is *0801*.

**User Response:** Refer to the section in the DB2 Replication Guide and Reference information about supported operating systems, versions, and releases for the replication architecture that was specified in the message.

---

**ASN1564E    The Replication action *action\_name* ended in error. The Capture server Replication architecture level for Capture schema *capture\_schema* does not support this Replication action.**

**Explanation:** The replication architecture level found in the `captureschema.IBMSNAP_REGISTER` does not allow the specified replication action.

**User Response:** Migrate the Capture server

control tables to 0801 before attempting this action.

---

**ASN1565E**    **The Replication action *action\_name* ended in error. The Apply Control Server Replication architecture level does not support this replication action.**

**Explanation:** The Replication architecture level found in the ASN.IBMSNAP\_SUBS\_SET does not allow the specified Replication action.

**User Response:** Migrate the Apply control server control tables to 0801 before attempting this action.

---

**ASN1567W**    **The table space container information for table space *tablespace\_name* cannot be read, because the DB2 stored procedure *procedure\_name* in the library *library\_name* cannot be found.**

**Explanation:** The DB2 Stored Procedure READTSCINFOS cannot be found on the Capture or target DB2 server. The stored procedure is required to retrieve DB2 table space container information for that server.

**User Response:** Determine whether the stored procedure exists on the server: check if the file db2rtsc exists in the function directory of the sqllib directory. The file db2rtsc might not exist if the server is a pre-V8 server. If the stored procedure does not exist, then edit the table space container definition provided in the output script.

---

**ASN1568E**    **The name length *length* for the DB2 object, *objectname* exceeds the allowed limit of *allowed\_limit*.**

**Explanation:** The DB2 object type provided in the second parameter allows a length that is smaller than the length of the actual object provided in the third parameter. As in the Properties file, the following values are valid for *object*: *Table*, *Index*, *Tablespace*, *Table owner*.

**User Response:** Refer to the DB2 SQL

Reference, and provide the correct name length.

---

**ASN1569E**    **The name of the DB2 object to be created is identical to the existing name *objectowner.objectname* of type *objecttype*.**

**Explanation:** The DB2 object cannot be created because there is already a DB2 object of the same type with the same name. As in the Properties file, the following values are valid for *object*: *Table*, *Index*, *Tablespace*, *Table owner*.

**User Response:** Provide a name for that object that does not already exist in DB2, and reissue the Replication task.

---

**ASN1570E**    **The DB2 object *object*, *objectowner.objectname* does not exist.**

**Explanation:** The DB2 object does not exist in the DB2 catalog. This object must exist in order to be defined as a source or target of a subscription set, as per the replication action. This object might have been defined as part of an existing registration or subscription-set definition but is not found in the DB2 catalog. As in the Properties file, the following values are valid for *object*: *Table*, *Index*, *Tablespace*, *Table owner*.

**User Response:** Provide a name that already exists in DB2, and reissue the Replication task. If the object was defined as part of an existing registration or subscription-set definition, verify that the object exists in the DB2 catalog.

---

**ASN1571E**    **The DB2 table *tableowner.tablename* cannot be created: the DB2 definition is not valid for data type *datatype* and column *column\_name*. The reason code is *reason\_code*.**

**Explanation:** The following values are valid for the *reason code*:

- 0        The datatype is not supported on this platform.
- 1        The length of the column is not supported on this platform.

- 2 The precision or scale of the column is not supported on this platform.

**User Response:** Refer to the SQL Reference for the appropriate DB2 platform.

---

**ASN1572E The row size *row\_size* for DB2 object *objectowner.objectname* of type *object\_type* exceeds its DB2 buffer pool row size *bufferpool\_rowsize*. The DB2 object cannot be created.**

**Explanation:** The row size of a table cannot exceed the table space page size for that table. The table space page size is derived from the buffer pool page size to which it belongs. No script is generated.

**User Response:** You might have to create the table in a different table space. Refer to your DB2 operating system documentation.

---

**ASN1573E The number of columns *number\_columns* for the DB2 object *objectowner.objectname* of type *object\_type* exceeds the DB2 limit *db2\_limit*. The DB2 object cannot be created.**

**Explanation:** The number of columns a DB2 object (table or index) can contain depends on the DB2 operating system but cannot exceed a predefined number. No script is generated. The following values are valid for object type: *table*, *index*.

**User Response:** Redesign the DB2 object.

---

**ASN1574E The DB2 PageSize *page\_size* for Tablespace *tablespace\_name* is not valid. Reason code *reason\_code*.**

**Explanation:** The PageSize must be valid for the table space to be created successfully. The following values are valid for reason code:

- 0 PageSize is not equal to the PageSize of the given buffer pool.
- 1 PageSize is not equal to one of the following: 4K, 8K, 16K, 32K.

**User Response:** Refer to the DB2 SQL Reference for appropriate PageSize ranges or values.

---

**ASN1575W The DB2 table *tableowner.tablename* will be created in the DB2 default Tablespace.**

**Explanation:** No table space name was specified indicating where to create the specified table, so the table will be created in the DB2 default table space. This might be a problem if the default table space specifications are not appropriate for the specified table.

**User Response:** Refer to the SQL Reference for the DB2 defaults. If you require the table to be in its own table space, then reissue the Replication task with the appropriate specifications. No action is required if the default is appropriate for the table.

---

**ASN1576W The DB2 index *index\_name* will be created in the DB2 default Indexspace or Tablespace.**

**Explanation:** A table space (for workstation operating systems) or an indexspace (for z/OS operating systems) was not provided into which the specified index might be created. Therefore, the index is created using the DB2 defaults. This might be a problem if the default specifications are not appropriate for the specified index.

**User Response:** Refer to the SQL Reference for the DB2 defaults. If you require the index to be in its own table space or indexspace, then reissue the Replication task with the appropriate specifications. No action is required if the default is appropriate for the index.

---

**ASN1577W The DB2 table space *tablespace* will be created in the DB2 default database.**

**Explanation:** For z/OS operating systems only, a database was not provided into which the specified table space might be created. Therefore, the table space is created using the DB2 defaults. This might be a problem if the default specifications are not appropriate for the specified table space.



**User Response:** Refer to the SQL Reference for the DB2 defaults. If you require the table space to be in its own database, then reissue the Replication task with the appropriate specifications. No action is required if the default is appropriate for the table space.

---

**ASN1578I**    **The DB2 table space *tablespace* will be created in the DB2 default storage group.**

**Explanation:** For workstation and z/OS operating systems only, a storage group was not provided into which the specified table space might be created. Therefore, the table space is created using the DB2 defaults. This might be a problem if the default specifications are not appropriate for the specified table space.

**User Response:** Refer to the SQL Reference for the DB2 defaults. If you require the table space to be in its own storage group, then reissue the Replication task with the appropriate specifications. No action is required if the default is appropriate for the table space.

---

**ASN1579I**    **The DB2 index *index\_name* will be created in the DB2 default storage group.**

**Explanation:** For workstation and z/OS operating systems only, a storage group was not specified into which the DB2 index might be created. Therefore, DB2 created the index using the default specification. This might be a problem if the default specifications are not appropriate for the specified index.

**User Response:** Refer to the SQL Reference for the DB2 defaults. If you require the index to be in its own storage group, then reissue the Replication task with the appropriate specifications. No action is required if the default is appropriate for the index.

---

**ASN1580I**    **The DB2 table space *tablespace* will be created in the DB2 default buffer pool.**

**Explanation:** For workstation and z/OS operating systems only, a buffer pool was not

provided into which the specified table space might be created. Therefore, the table space is created using the DB2 defaults. This might be a problem if the default specifications are not appropriate for the specified table space.

**User Response:** Refer to the SQL Reference for the DB2 defaults. If you require the table space to be in its own buffer pool, then reissue the Replication task with the appropriate specifications. No action is required if the default is appropriate for the table space.

---

**ASN1581I**    **The DB2 index *index\_name* will be created in the DB2 default buffer pool.**

**Explanation:** For workstation and z/OS operating systems only, a buffer pool was not provided into which the specified index might be created. Therefore, the index is created using the DB2 defaults. This might be a problem if the default specifications are not appropriate for the specified index.

**User Response:** Refer to the SQL Reference for the DB2 defaults. If you require the index to be in its own buffer pool, then reissue the Replication task with the appropriate specifications. No action is required if the default is appropriate for the index.

---

**ASN1582W**    **The Tablespace *tablespace* will be created in Buffer Pool *buffer\_pool* but the buffer pool does not exist or is not active.**

**Explanation:**

- For applications on a DB2 UDB database, the buffer pool does not exist into which the specified table space might be created.
- For applications on a DB2 for z/OS database, the buffer pool is not active into which the table space might be created.

**User Response:**

- For the DB2 UDB database, make sure that the buffer pool exists at the time of running the script.

- For a DB2 for z/OS database, make sure the buffer pool is active at the time of running the script.
- 

---

**ASN1583E**    **The PageSize *page\_size* for Tablespace *tablespace* does not match the default buffer pool PageSize.**

**Explanation:** The given PageSize does not match the PageSize of the default buffer pool. The table space cannot be created.

**User Response:** Change the PageSize or choose another buffer pool.

---

**ASN1584E**    **The Replication action *action\_name* ended in error. The Capture server Replication architecture level *arch\_level* for Capture schema *capture\_schema* is not a valid architecture level.**

**Explanation:** The Replication architecture level found in the captureschema.IBMSNAP\_REGISTER does not allow the specified Replication action.

**User Response:** Drop the control tables on the Capture control server manually because the architecture level is not supported. Create the control tables with a valid architecture level.

---

**ASN1585E**    **The Replication action *action\_name* ended in error. The Apply control server Replication architecture level *arch\_level* is not a valid architecture level.**

**Explanation:** The Replication architecture level found in the ASN.IBMSNAP\_SUBS\_SET does not allow the specified Replication action.

**User Response:** Drop the control table on the Apply control server manually because the architecture level is not supported. Create the control tables with a valid architecture level.

---

**ASN1586W**    **The DB2 table *tableowner.tablename* will be created in the DB2 default database.**

**Explanation:** For z/OS operating systems only, a database was not provided into which the specified table might be created. Therefore, the table is created using the DB2 defaults. This might be a problem if the default specifications are not appropriate for the specified table.

**User Response:** Refer to the SQL Reference for the DB2 defaults. If you require the table space to be in its own database, then reissue the Replication task with the appropriate specifications. No action is required if the default is appropriate for the table.

---

**ASN1587E**    **The value *value* for the parameter *parameter\_name* of the DB2 object *db2object\_name*, which has a type of *type*, is not valid.**

**Explanation:** The provided value is not valid or conflicts with another parameter value.

**User Response:** Refer to the SQL Reference for valid values.

---

**ASN1588E**    **The value *encoding\_scheme* provided for the parameter *encoding\_scheme* is not valid for the DB2 server *server\_name*.**

**Explanation:** The provided value for the encoding scheme is not valid for the DB2 version of the server. No script is generated.

**User Response:** Refer to the SQL Reference for a valid value of the encoding schema for the DB2 version.

---

**ASN1589W** The calculation of the size of the table space container *container* of the table space *tspc* resulted in an incorrect container size. Therefore the container size has been changed to size *size* megabytes.

**Explanation:** The calculation of the table space container size has resulted in a value that is too low to be used in a valid table space container definition. To ensure that the definition will be accepted by DB2, a replication specific minimum container size has been provided for the table space container definition.

**User Response:** For the calculation based on a percentage of the current source table size, check whether the source table contains data and if the statistics of the source table are up to date (using the RUNSTATS utility) . For the calculation based on a number of rows, check whether the number of rows is realistic.

---

**ASN1590E** The DB2 table space *table\_sp\_name* is partitioned and in the DB2 *object\_type* group. It should not be partitioned and it should be in the *object\_type* IBMCATGROUP.

**Explanation:** The provided table space is a partitioned table space and does not reside on the DB2 catalog node or partition group. Creation of the Replication control tables in a partitioned table space is not supported. No Script is generated.

**User Response:** Specify a table space that is not partitioned.

---

**ASN1600E** The REMOTE SERVER *remote\_servername* cannot be found.

**Explanation:** The specified remote server name cannot be found in the Federated Catalog table SYSIBM.SYSSERVERS, column REMOTE\_SERVER. The non-DB2 relational server cannot be accessed.

**User Response:** Verify the input provided for the remote server name and call the action again,

or populate the Federated Catalog table appropriately.

---

**ASN1601E** The REMOTE AUTHID information for the REMOTE SERVER *remote\_servername* cannot be found.

**Explanation:** The remote authentication information cannot be found in the Federated Catalog table SYSIBM.SYSXXXXX, for the REMOTE\_SERVER value provided. The non-DB2 relational server cannot be accessed.

**User Response:** Verify the input provided for the remote server name and call the action again, or populate the Federated Catalog table appropriately.

---

**ASN1602E** The server *server\_alias* does not support access to Federated servers.

**Explanation:** The Federated replication functions are only supported on the DB2 UDB Workstation V8 and higher.

**User Response:** Make sure the specified database server is one of those listed above or do not issue the replication task against a server that does not support it.

---

**ASN1603E** The Replication Apply control server cannot reside on a non-IBM server.

**Explanation:** Non-DB2 relational servers can be replication Capture control servers or target servers, but they cannot be Apply control servers.

**User Response:** Specify a DB2 server as the Replication Apply control server.

---

**ASN1604E The remote table**  
*remoteowner.tablename exists in the non-IBM server, but the provided nickname nicknameowner.nickname cannot be found in the Federated server.*

**Explanation:** The specified remote table exists in the remote database but the corresponding nickname is not found in the federated database.

**User Response:**

1. Refer to the DB2 Federated manuals on how to create a nickname.
2. Create the nickname in the federated database.
3. Issue the Replication task again.

---

**ASN1605E The nickname**  
*nicknameowner.nickname exists in the Federated server but the remote table remoteowner.remotetable cannot be found in the non-IBM server.*

**Explanation:** The nickname for the specified remote table exists but the corresponding remote table does not exist in the remote database. This is an inconsistent state of definitions when creating Replication definitions.

**User Response:**

1. Drop the nickname.
2. Depending on the table type, perform the following actions:
  - If the table is a user table, create the remote table in the remote server.
  - If the table is a replication control table on the Capture control server, perform the following actions:
    - a. Copy the data from the existing control tables on the Capture control server.
    - b. Drop the control tables on the Capture control server.
    - c. Create the control tables on the Capture control server.
3. Create the Nickname in the federated server.
4. Issue the Replication task again.

---

**ASN1606W The nickname**  
*nickname\_owner.nickname\_name exists in the Federated server but the remote table table\_owner.table\_name cannot be found in the non-IBM server.*

**Explanation:** The nickname for the specified remote table exists but the corresponding remote table does not exist in the remote database. Although this is an orphan nickname, this inconsistent state is still tolerated when dropping Replication definitions. A script is generated.

**User Response:** The source nickname is not dropped when dropping the replication definitions. To ensure a consistent catalog, drop the nickname.

---

**ASN1607W It is strongly recommended to alter the nickname**  
*nickname\_owner.nickname\_name defined for the Replication subscription target: to alter the local data type of column column\_name from existing\_local\_datatype to recommended\_local\_datatype and ensure the proper source to target column data type mapping.*

**Explanation:** A mismatch was found between a source column data type and its corresponding nickname target column data type, that does not violate DB2 compatibility rules, but that might cause a problem to native non-IBM end-user applications. The problem does not occur during replication of the column data. The problem does occur if end-user applications retrieve the data. For example, if the nickname data type is created using the default mappings from the non-DB2 relational data type to the DB2 data type, the column will hold the broadest range of data type values, which might clash with the end-user application requirement of a more restrictive data type. A script is generated.

**User Response:** Check the target to ensure that the nickname data type you need at the target is indeed the source column data type. If it is, then issue an 'ALTER NICKNAME' statement to

change the local data type of the nickname column. When you alter the nickname local data type to be the same as the source column data type, you enforce that the end-user application on the non-DB2 relational server sees the same data type as the source column data type.

---

**ASN1608I**    **The nickname *source\_nickname* for the source and the nickname *ccd\_nickname* for the Consistent Change Data table have a column data type that is altered: the local data type column *local\_datatype* is set to *changed\_datatype* because the remote data type is *remote\_datatype*. Reason Code *reason\_code*.**

**Explanation:** This message is issued at the time of creating the nickname for the CCD. The nickname is altered based on the data type of the CCD created in the non-DB2 relational server, to ensure the proper data type setting. Failure to do so would result in improper Replication behavior. A script is generated, that updated user-provided definitions.

**User Response:** No action is required if the Replication updates are acceptable.

---

**ASN1609E**    **The nickname *nicknameowner.nickname* exists in the Federated server but the remote table *remoteowner.remotetable* does not contain all the necessary columns.**

**Explanation:** The target table nickname exists, and contains only a subset of the columns requested in the subscription.

**User Response:** Use another Nickname as the target table or change the subscription to match the columns in the existing nickname.

---

**ASN1620E**    **Both Capture server control tables and Apply control server control tables already exist. Capture server control tables exist with architecture level *capturearch\_level* and Capture Schema *capture\_schema*. Apply control server control tables exist with architecture level *applyarch\_level*.**

**Explanation:** The tables `captureschema.IBMSNAP_REGISTER` and `captureschema.IBMSNAP_SUBS_SET` already exist at the given server.

**User Response:**

- If the architecture level of the existing `captureschema.IBMSNAP_REGISTER` control table is *0201*:
  - if the existing `captureschema.IBMSNAP_REGISTER` is already populated with valid Replication definitions, migrate the Capture server control tables to the *0801* architecture,
  - if the table is empty, simply drop the pre-V8 Capture server control tables and reissue the Replication task again.
- If the architecture level of the existing control tables is not *0201*, consider creating Capture server control tables with a different capture schema name.
- If the architecture level of the existing `IBMSNAP_SUBS_SET` control table is *0201*:
  - if the existing `ASN.IBMSNAP_SUBS_SET` table is already populated with valid Replication definitions, migrate the Apply control server control tables to the *0801* architecture,
  - if the table is empty, simply drop the pre-V8 Apply control server control tables and reissue the Replication task again
- If the architecture level of the existing control tables is not *0201*, consider creating Apply control server control tables on a different server.

---

**ASN1621W At least one row was found in the control table**

*controlowner:controltable. Dropping of this control table will result in dropping of all Replication definitions stored in the table.*

**Explanation:** The control tables that are dropped are not empty. Replication control information is deleted if the generated scripts are executed.

**User Response:** Run the generated scripts only if you can ensure the following dependencies:

- Understand the impact to existing dependent subscription sets of dropping the control tables from the Capture control server.
- Understand the impact to existing dependent subscription sets (for multi-tier scenarios) of dropping control tables from the Apply control server.
- You do not want Replication to run the Capture or the Apply processes for these definitions anymore.

If the architecture level is *0201*, migrate the Capture or Apply control server control tables to the *0801* architecture before dropping the control tables.

---

**ASN1622E The Replication action *action\_name* ended in error. The required control table *controlowner:controltable* could not be found.**

**Explanation:** Replication definitions are stored in Replication control tables. These tables must exist before a registration or subscription definition can be created. The existence of the IBMSNAP\_REGISTER table is used to check if the control tables for the Capture control server already exist for a particular Capture schema. The existence of the IBMSNAP\_SUBS\_SET table is used to check if the control tables on the Apply control server already exist. The existence of IBMSNAP\_SUBS\_MEMBR is checked at the time of checking for the existence of a subscription member.

**User Response:** If the control table

IBMSNAP\_SUBS\_MEMBR table does not exist, then your environment is in an inconsistent state. You must drop all the control tables from the Apply control server and then create them before attempting the action.

Alternatively, if the control tables IBMSNAP\_REGISTER or IBMSNAP\_SUBS\_SET do not exist, create them before adding registration or subscription definitions on a control server. Otherwise, you can do the following:

1. If you are doing a registration-related action, check if the appropriate Capture schema was provided; or if the appropriate Capture control server was provided as input.
2. If you are doing a subscription-related action, check if the appropriate Apply control server was provided as input.
3. If you are creating a subscription set that contains target tables that need to be auto-registered at the target server (CCD or replica), then check if the appropriate control tables for the Capture control server exist at the subscription target server.

---

**ASN1623W The Replication control table, *controlowner:controltable* could not be found and is not dropped.**

**Explanation:** The *Drop Capture control tables* or *Drop Apply control server control tables* action was issued and the control table was missing. The script will not generate the appropriate DROP statement for that control table.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1624I The server *server\_alias* is not a known Replication Capture server for *capture\_schema*.**

**Explanation:** The *captureschema.IBMSNAP\_REGISTER* table could not be found. A server is defined as a Replication Capture server when the appropriate Capture server control tables (including the IBMSNAP\_REGISTER table) exist on the server.

**User Response:** Create the appropriate Capture

server control tables, if needed.

---

**ASN1625I**    **The server *server\_alias* is not a known Replication Apply control server.**

**Explanation:** The ASN.IBMSNAP\_SUBS\_SET table could not be found. A server is defined as a Replication Apply control server when the appropriate Apply control server control tables (including the IBMSNAP\_SUBS\_SET table) exist on the server.

**User Response:** Create the appropriate control tables on the Apply control server, if needed.

---

**ASN1626E**    **Capture server control tables already exist for architecture level *arch\_level* with the same Capture schema.**

**Explanation:** The table captureschema.IBMSNAP\_REGISTER already exists at the given server.

**User Response:**

- If the architecture level of the existing captureschema.IBMSNAP\_REGISTER table is *0801*, consider the following options:
  - Running the command is not necessary because the tables already exist with the same Capture schema.
  - Run the command under a different Capture schema.
- If the architecture level of the existing captureschema.IBMSNAP\_REGISTER control table is *0201*:
  - Migrate the Capture control server control tables to the *0801* architecture, if the existing captureschema.IBMSNAP\_REGISTER is already populated with valid Replication definitions.
  - If the control table is empty, simply drop the pre-V8 Capture server control tables and issue the Replication task again.

Otherwise, the architecture level is not valid. You need to drop the tables manually before attempting to create the tables.

---

**ASN1627E**    **Some Capture server control tables already exist with the same Capture Schema but for which an architecture level cannot be determined.**

**Explanation:** The table captureschema.ASN.IBMSNAP\_REGISTER does not exist although other Capture server control tables were found at the given server. Capture server control tables cannot be created until the tables are dropped. The Replication definitions at the Capture server are in an inconsistent state.

**User Response:** Drop the remaining Capture server control tables to clean up the Capture control server definitions, and reissue the Create control table task. Loss of data occurs, so look at the content of the remaining control tables before issuing the drop task.

---

**ASN1628E**    **The Capture server control tables are not at the architecture level requested.**

**Explanation:** The table captureschema.IBMSNAP\_REGISTER does not exist with the provided architecture level. No script is generated.

**User Response:** Issue the replication task again at the appropriate architecture level for the appropriate Capture control server and Capture schema.

---

**ASN1629E**    **No Capture server control tables were found for the provided Capture schema.**

**Explanation:** No control tables exist on the Capture control server. No control tables are dropped, and no script is generated.

**User Response:** Issue the replication task again at the appropriate architecture level for the appropriate Capture control server and Capture schema.

---

**ASN1630W** Some Capture server control tables already exist with Capture Schema *capture\_schema* but their architecture level cannot be determined. The Replication action *action\_name* for the provided architecture level *arch\_level* and Capture Schema will drop control tables that might not belong to the architecture level provided.

**Explanation:** The table captureschema.IBMSNAP\_REGISTER does not exist on the Capture server. The Replication architecture level is unknown, and if you provide an incorrect architecture level, you might lose critical data. No checks occur to determine whether a particular Capture server control table architecture level can be inferred. The control table is dropped if it exists. A script is generated.

**User Response:** Issue the task again with the appropriate architecture level for DB2 replication.

---

**ASN1631E** Apply control server control tables already exist for architecture level *arch\_level*.

**Explanation:** The table ASN.IBMSNAP\_SUBS\_SET already exists at the given server. No script is generated.

**User Response:** If the architecture level of the existing ASN.IBMSNAP\_SUBS\_SET control table is *0201*:

- If the existing ASN.IBMSNAP\_SUBS\_SET is already populated with valid Replication definitions, Apply control server control tables to the *0801* architecture,
- If the table is empty, simply drop the pre-V8 Apply control server control tables and reissue the Replication task again.

Otherwise, the architecture level is not valid. You need to drop the tables manually before attempting to create the tables.

---

**ASN1632E** Some Apply control server control tables already exist but for which an architecture level cannot be determined.

**Explanation:** The table ASN.IBMSNAP\_SUBS\_SET does not exist although other Apply control server control tables were found at the given server. Apply control server control tables cannot be created until the tables are dropped. The Replication definitions at the Apply control server are in an inconsistent state. No script is generated.

**User Response:** Drop the remaining control tables on the Apply control server to clean up the Apply control server replication definitions. Reissue the *Create control table* task. Loss of data occurs, so look at the content of the remaining control tables before issuing the *Drop* task.

---

**ASN1633E** The Apply control server control tables are not at the architecture level requested.

**Explanation:** The table ASN.IBMSNAP\_SUBS\_SET does not exist with the provided architecture level. No script is generated.

**User Response:** Issue the replication task again at the appropriate architecture level for the appropriate Apply control server.

---

**ASN1634E** No Apply control server control tables were found.

**Explanation:** There are no control tables to drop from the Apply control server. No script is generated.

**User Response:** Issue the replication task again at the appropriate architecture level for the appropriate Apply control server.



---

**ASN1635W** Some Apply control server control tables already exist but their architecture level cannot be determined. The Replication action *action\_name* for the provided architecture level *arch\_level* will drop control tables that might not belong to the architecture level provided.

**Explanation:** The table ASN.IBMSNAP\_SUBS\_SET does not exist on the Apply control server. The Replication architecture level is unknown, and if you provide an incorrect architecture level, you might lose critical data. No checks occur to determine whether a particular Apply control server control table architecture level can be inferred. If the control table exists, it is dropped. A script is generated.

**User Response:** Reissue the task with the appropriate architecture level for DB2 replication.

---

**ASN1636E** The Replication Action of Manual Full Refresh ended with an error for the Apply qualifier *apply\_qual* and set name *set\_name*. The synchpoint in the *capschema.IBMSNAP\_PRUNCNTL* table for the source member *sourceowner.sourcetable* and the target member *targetowner.target\_table* is not translated by the Capture program.

**Explanation:** The synchpoint is either less than 0 or equal to hex zeros.

**User Response:** Make sure you run the before load script to translate the hex zeros and capture is running on the server.

---

**ASN1637E** The replication action 'Manual Full Refresh' ended in error for the Apply qualifier *apply\_qualifier* and the set name *set\_name*. The target structure of at least one of the target subscription-set members in the given subscription set is greater than eight. None of the subscription-set members is eligible for a manual full refresh.

**Explanation:** The target structure of at least one of the target subscription-set members in the given subscription set is greater than eight. A manual full refresh does not support target structures that are greater than eight.

**User Response:** Make sure that the target structure of the subscription-set member is less than or equal to eight, and then reissue the replication task.

---

**ASN1638W** The subscription-set member with a target of *targetowner.targetname* and a source of *sourceowner.sourcename* is not complete. This subscription-set member is not included in the manual full refresh.

**Explanation:** The manual full refresh supports complete targets only. The given subscription-set member is not complete and cannot be included.

**User Response:** No action is required.

---

**ASN1639E** The replication action 'Manual Full Refresh' ended in error for the Apply qualifier *apply\_qualifier* and the set name *set\_name*. None of the target subscription-set members in the given subscription set is complete or eligible for a manual full refresh.

**Explanation:** The manual full refresh supports complete targets only, and none of the targets is complete.

**User Response:** Make sure that at least one of

the subscription-set members in the subscription set is complete, and reissue the replication task.

---

**ASN1640E**    **The replication action ended in error for the Apply qualifier *apply\_qualifier* and the set name *set\_name*. There are no subscription-set members in the subscription set.**

**Explanation:** The subscription set does not contain any subscription-set members.

**User Response:** Add at least one subscription-set member to the subscription set, and reissue the replication task.

---

**ASN1641E**    **The replication action *action\_name* ended in error. This action on an OS/400 system is supported only through OS/400 commands.**

**Explanation:** Neither the replication center nor the command line supports the replication action on an OS/400 system. The possible actions might be: creating capture server control tables, creating apply server control tables, dropping capture server control tables, or dropping apply server control tables.

**User Response:** Issue OS/400 commands to perform the replication action.

---

**ASN1650I**    **The replication action *action\_name* started at *timestamp*. The monitor server is *server\_name* and the Group\_or\_Contact name is *group\_name\_or\_contact\_name*.**

**Explanation:** The replication action started at the specified monitor server.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1653I**    **The replication action *action\_name* for *group\_contact\_or\_condition\_name* ended successfully at *timestamp*. The monitor server is *server\_name*.**

**Explanation:** The replication action ended successfully at the specified monitor server.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1654E**    **The replication action *action\_name* ended in error. The length of the input parameter *parameter-name*, *parameter\_length* exceeds the limit *maximum-limit*.**

**Explanation:** The length of the specified input parameter is longer than the maximum allowable length. No script is generated.

**User Response:** Verify the input parameter value, and re-enter the parameter value.

---

**ASN1655E**    **The replication action *action\_name* ended in error. The value *input\_value* of the input parameter *input\_parameter* is incorrect.**

**Explanation:** The value of the specified input parameter is not correct.

**User Response:** Refer to your documentation for valid parameter values.

---

**ASN1656E**    **The replication action *action\_name* ended in error. The value of the input parameter *input\_parameter* is missing.**

**Explanation:** A value for this specified input parameter is mandatory for this action. However, the value is missing. No script is generated.

**User Response:** Enter a value for this mandatory input parameter, and rerun the replication action.

---

**ASN1657E**    **The replication action *action\_name* ended in error. At least one optional parameter value must be specified.**

**Explanation:** You must specify at least one optional parameter value when issuing a command in which each parameter value is optional. No script is generated.

**User Response:** Issue the command again with the correct parameters.

---

**ASN1658E**    **The replication action *action\_name* ended in error. The value *value1* of the input parameter *input\_parameter1* must be different than the value *value2* of the input parameter *input\_parameter2*.**

**Explanation:** The value of one input parameter is the same as the value of another input parameter and will result in the creation of inconsistent definitions. No script is generated.

**User Response:** Issue the command again with valid parameter values.

---

**ASN1659E**    **The replication action *action\_name* ended in error. The contact *contact-name* already exists.**

**Explanation:** The specified contact name already exists in one of the rows in the ASN.IBMSNAP\_CONTACTS table. Contact names must be unique. No script is generated.

**User Response:** Issue the command again with a different contact name.

---

**ASN1660E**    **The replication action *action\_name* ended in error. The contact *contact-name* does not exist.**

**Explanation:** The specified contact name does not exist in any of the rows in the ASN.IBMSNAP\_CONTACTS table. The contact name must exist in the ASN.IBMSNAP\_CONTACTS table before you can alter, substitute, delegate, or drop the name. No script is generated.

**User Response:** Issue the command again with a different contact name.

---

**ASN1661E**    **The replication action *action\_name* ended in error. The contact *contact-name* cannot be dropped, because dropping the contact empties each associated group.**

**Explanation:** A group should have at least one associated contact. The specified contact is the last contact in each associated group, and the last contact cannot be dropped. No script is generated.

**User Response:** Drop each associated group before attempting to drop the contact.

---

**ASN1662E**    **The replication action *action\_name* ended in error. The contact *contact-name* cannot be dropped, because the contact is associated with one or more conditions.**

**Explanation:** The contact name that you are attempting to drop is the only contact associated with conditions for either the Capture or Apply components. No script is generated.

**User Response:** Use the SUBSTITUTE option in the DROP CONTACT command, or use the SUBSTITUTE command to change the contact name of the conditions. If you do not need the conditions, drop the conditions and then drop the contact.

---

**ASN1663E**    **The replication action *action\_name* ended in error. The value *startdate\_value* that is specified for the start date is greater than the value *enddate\_value*, which is specified for the end date.**

**Explanation:** You cannot enter a start date that is beyond the end date. No script is generated.

**User Response:** Issue the command again with a valid combination of dates.

---

**ASN1664E**    **The replication action *action\_name* ended in error. The group *group-name* already exists.**

**Explanation:** The specified group name already exists in one of the rows in the ASN.IBMSNAP\_GROUPS table. Group names must be unique.

**User Response:** Change the group name, and issue the command again.

---

**ASN1665E**    **The replication action *action\_name* ended in error. The group *group\_name* does not exist.**

**Explanation:** The specified group name does not exist in any of the rows in the ASN.IBMSNAP\_GROUPS table. The group name must exist in the ASN.IBMSNAP\_GROUPS table before you can alter or drop the group name. No script is generated.

**User Response:** Verify the group name, and reissue the command.

---

**ASN1666E**    **The replication action *action\_name* ended in error. The group *group\_name* cannot be dropped because it is associated with one or more conditions.**

**Explanation:** The group that you are attempting to drop is the only group associated with conditions for either the Capture or Apply components. No script is generated.

**User Response:** In order to drop the group, alter the contacts of the associated conditions and then reissue the command.

---

**ASN1667E**    **The replication action *action\_name* ended in error. The contact *contact-name* is not associated with the specified group *group\_name*.**

**Explanation:** The contact name that you are attempting to drop is not associated with the specified group.

**User Response:** Verify the specified contact name and reissue the command.

---

**ASN1668E**    **The replication action *action\_name* ended in error. The contact *contact-name* is already associated with the specified group *group\_name*.**

**Explanation:** The contact name that you specified is already associated with the specified group.

**User Response:** No action is required.

---

**ASN1671E**    **The replication action *action\_name* ended in error. The alert condition *condition-name* already exists for the monitor qualifier *mon-qual*, the server *server-name*, the schema or qualifier *schema-or-qualifier*, and the subscription-set name *set-name*.**

**Explanation:** The alert condition that you are attempting to create already exists with the same specified parameters on the monitor control server.

**User Response:** Verify this alert condition and issue the command again.

---

**ASN1672E**    **The replication action *action\_name* ended in error. The alert condition *condition-name* does not exist for the monitor qualifier *mon-qual*, the server *server-name*, the schema or qualifier *schema-or-qualifier*, and the subscription-set name *set-name*.**

**Explanation:** The alert condition that you are attempting to drop or to alter does not exist on the monitor control server.

**User Response:** Verify the alert name and issue the command again.

---

**ASN1673W**    **The condition *condition\_name* is valid only at the apply qualifier level.**

**Explanation:** The condition name is not valid with a subscription-set name value. The name of

the subscription set will be ignored.

**User Response:** Do not specify the subscription-set name value.

---

**ASN1674W**    **The condition *condition\_name* is valid only with update-anywhere subscription sets.**

**Explanation:** The condition name is valid only with update-anywhere subscription sets.

**User Response:** Do not set this condition. This condition will be ignored.

---

**ASN1675I**    **This is a test message from the Replication Center.**

**Explanation:** This message is used to send a test e-mail verifying the e-mail address entered in the contact.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1677E**    **The replication action *action\_name* ended in error. The apply qualifier *apply-qual* and the subscription-set name *set-name* do not exist on the server *server-name*.**

**Explanation:** The apply qualifier and the subscription-set name do not exist in the ASN.IBMSNAP\_SUBS\_SET table on the specified apply control server.

**User Response:** Supply a valid apply qualifier and a valid subscription-set name.

---

**ASN1678E**    **The replication action *action\_name* ended in error. The capture schema *cap-schema* does not exist on the server *server-name*.**

**Explanation:** The capture schema does not exist in the ASN.IBMSNAP\_CAPSCHEMAS table on the specified capture control server.

**User Response:** Supply a valid capture schema.

---

**ASN1679E**    **The replication action *action\_name* ended in error. The contact *contact\_name* that you attempted to substitute is not associated with a condition.**

**Explanation:** The contact name does not exist in the ASN.IBMSNAP\_CONDITIONS table. A contact can be substituted only if it exists in the ASN.IBMSNAP\_CONDITIONS table. No script is generated.

**User Response:** Supply a valid contact name.

---

**ASN1680I**    **The Replication action *action\_name* started at *time*. The Monitor server is *server\_name*.**

**Explanation:** This message is for your information only.

**User Response:** No action required.

---

**ASN1681E**    **The Replication action ended in error. Monitor control tables already exist for Architecture *Levelarch\_level*.**

**Explanation:** The monitor control tables already exists at the monitor server.

**User Response:** If the architecture level of the existing monitor tables is 0801, there is no need to run the command since the tables already exist.

---

**ASN1682E**    **The Replication action ended in error. No Monitor control tables were found.**

**Explanation:** There are no Monitor control tables to drop. No script will be generated.

**User Response:** Issue the replication task again for the appropriate server containing the Monitor control tables.

---

**ASN1700E    The column**  
*tableowner.tablename.columnname* of  
**data type *data\_type* cannot be**  
**included in the registration.**  
**Reason Code** *reason\_code*.

**Explanation:** The column cannot be supported by the Replication Capture mechanism, as defined. No script for the registration of the column specified is generated. The following values are valid for the reason code:

- 0**        The data type is not supported.
- 1**        The column is already registered.
- 2**        z/OS fieldproc column.
- 3**        This column does not qualify as a before-image column.
- 4**        The data type is not supported through DB2 for federated.
- 5**        The column does not exist in the source object.
- 6**        The maximum number of registered LOB columns was exceeded for that table.
- 7**        The column name starts with the before-image prefix.
- 8**        This column does not qualify as a before-image column or as an after-image column.

**User Response:** Check the Reason Code to get the reason why the column cannot be registered. Refer to the DB2 Replication Guide and Reference for additional explanations or restrictions.

---

**ASN1701E    The provided locksize value**  
*lock\_size* for the given table space  
*tablespace\_name* is not valid.

**Explanation:** Locksize should be equal to the P(PAGE), R(ROW) or A(ANY), in the case of z/OS operating system.

**User Response:** Provide the correct locksize and submit your action again.

---

**ASN1702W    Replication definitions for the**  
**registered column**  
*objectowner.objectname.columnname*  
**has been changed to support null**  
**values.**

**Explanation:** before-image columns are required to support null values. If no before-image column value is present, an INSERT statement will fail. A script is generated to update user-provided definitions.

**User Response:** This message is for your information only; no action is required.

---

**ASN1703E    The table *tableowner.tablename***  
**cannot be registered for**  
**change-capture replication.**  
**Reason code** *reason\_code*.

**Explanation:** The table cannot be supported by the Replication Capture mechanism, as defined. No script is generated. The following values are valid for the reason code:

- 0**        The table with a z/OS validproc.
- 1**        Existing internal CCD table.
- 2**        Existing CD table.
- 3**        DB2 catalog table (Windows, UNIX, iSeries)
- 4**        The table is already registered.
- 5**        The source for an internal CCD table is not a registered source.
- 6**        The source is a CD table and cannot be registered.
- 7**        This source name is a duplicate for this session.
- 8**        The source is a replication control table.
- 9**        Not one of the source columns qualifies for registration.
- 10**       The maximum number of registered LOB columns has been exceeded for this table.
- 11**       Structured data types are not supported.

12 The before-image prefix can be only one character.

13 An internal error occurred.

**User Response:** Check the Reason Code to get the reason why the table cannot be registered for change-capture replication. Refer to the DB2 Replication Guide and Reference for additional explanations and restrictions.

---

**ASN1704E** The view *viewowner.viewname* cannot be registered. Reason code *reason\_code*.

**Explanation:** The view cannot be supported by the Replication Capture mechanism, as defined. No script is generated. The following values are valid for the reason code:

- 0 None of the dependent tables for the view are registered.
- 1 The source-table columns on which the view is dependent are not registered.
- 2 The view is on an internal ccd.
- 3 The view is already registered.
- 4 The view has an 'OUTER JOIN' syntax.
- 5 The view includes more than one table or view column with a function, and no correlation is provided in the view definition for each table.
- 6 The view contains a reference to an aggregate function.
- 7 The view contains a subselect/subquery.
- 8 The view contains a reference to another view.
- 9 The view has an UNION.
- 10 No correlation is provided for the column.
- 11 The base table does not have the schema name.
- 12 The base table does not exist.
- 13 The view contains Table Expression as Table.

14 The dependent table does not exist.

15 A view on view cannot be registered.

16 The given source object is not a view.

17 This source view is a duplicate for this session.

18 The view definition cannot be supported.

19 The view has an asterisk (\*) instead of a specific column name in the view definition.

20 The view contains the join of a CCD and a non-CCD table.

**User Response:** Check the Reason Code to get the reason why the view cannot be registered. Refer to the DB2 Replication Guide and Reference for additional explanations and restrictions.

---

**ASN1705E** The change data *object*, *objectowner.objectname* already exists in the server.

**Explanation:** The change data table or view cannot be used for the current source to be registered, because it already exists at the Capture server. No script is generated.

**User Response:** Provide a different name for the change data object.

---

**ASN1706W** A column *column\_name* is added to a registered source *sourceowner.sourcename*. The registered source maintains an Internal CCD table. The new column has to be first added to the CCD table subscription member before adding to any existing or not-yet existing subscription member.

**Explanation:** If the new column is needed in dependent subscription sets, you first have to add the column to the internal CCD subscription member before adding the column to any subscription member needed.

**User Response:** Provide a different name for the change data object.

---

**ASN1707W The Replication action Alter Registration for *sourceowner.sourcename* is not in effect until a Capture REINIT command is issued at the Capture server.**

**Explanation:** The registered source is successfully updated. However, the Capture program does not recognize the corresponding captureschema.IBMSNAP\_REGISTER table updates until a REINIT command forces it to do so. A script is generated. A Capture command is required afterwards for the effect of the script to be in action.

**User Response:** To make the changes effective immediately:

1. Run the generated script.
2. Issue a REINIT of the appropriate Capture program, for the appropriate Capture schema.

---

**ASN1708E The table, view or nickname *objectowner.objectname* is not a Replication registered source.**

**Explanation:** The Replication object specified above is not defined in the replication control tables. No script is generated.

**User Response:** Ensure that the object is correctly specified in the command and it exists.

---

**ASN1709W Associated subscription sets will not be valid after the registered source *sourceowner.sourcename* is dropped.**

**Explanation:** Subscription members rely on the underlying source registrations that define the source member. If you drop a registered source table, the dependent source members of a subscription set are no longer valid. The subscription sets that are associated with the specified registration source can be found in the captureserver.IBMSNAP\_PRUNCNTL table at the Capture control server, where the

SOURCE\_OWNER and SOURCE\_TABLE correspond to the registered source that is dropped. The appropriate Apply control server and subscription set names are columns in the IBMSNAP\_PRUNCNTL table. The associated subscription sets fail if Apply is running. A script is generated.

**User Response:** Deactivate or drop the dependent subscription sets before running the script, if the registered source has dependent subscription sets.

---

**ASN1710W Dependent view registered sources will not be valid after the registered source *sourceowner.sourcetable* is dropped.**

**Explanation:** View registrations rely on the underlying registration of the tables that make up the view definition. If you drop a registered source table, you invalidate any view registration that is based on the table. The views that might be affected can be found in the captureserver.IBMSNAP\_REGISTER table at the Capture server, where the PHYS\_CHANGE\_OWNER and PHYS\_CHANGE\_TABLE are the same as the CD\_OWNER and CD\_TABLE of the registered source that is dropped. The associated subscription sets, which depend on the view registrations, fail, if Apply is running. A script is generated.

**User Response:** Deactivate or drop the appropriate subscription sets, or view registrations, before running the script, if the registered source has dependent view registrations.

---

**ASN1711W The source *sourceowner.sourcename* is still active so dropping it will result in a Capture failure.**

**Explanation:** An active registration has a SYNCHPOINT value that is not null in its captureschema.IBMSNAP\_REGISTER table. When the Capture program started, it expected all active registrations to always exist and be valid. So the Capture program needs to be signaled that a registered source was dropped



because the drop action invalidates the registration information. Failure to provide that information to the Capture program causes the Capture program to fail. A script is generated, but is NOT ready to run.

**User Response:**

1. Deactivate the appropriate registration (via the Replication Center GUI, or by issuing the STOP signal and a command type of CMD).
2. Wait for a SIGNAL\_STATE of Complete in the captureschema.IBMSNAP\_SIGNAL table.
3. Run the script that drops the registration.

---

**ASN1712E    The table, view, or nickname *objectowner.objectname* is not a valid Replication registered source. Reason code *reason\_code*.**

**Explanation:** Inconsistent information was found for this registered source in the Capture server control tables. No script is generated.

**User Response:** Drop the registered source and create the registration again.

---

**ASN1713E    The registered source *sourceowner.sourcename* cannot be deactivated. Reason code *reason\_code*.**

**Explanation:** The following values are valid for reason code:

- |          |   |
|----------|---|
| <b>0</b> | The source is registered as a FULL REFRESH and therefore cannot be deactivated. |
| <b>1</b> | The source is a CCD and CCD registrations cannot be deactivated.                |
| <b>2</b> | The source is a view and view registrations cannot be deactivated.              |

**User Response:** This message is for your information only, and no action is required.

---

**ASN1714E    The registered source *sourceowner.sourcename* cannot be altered. Reason code *reason\_code*.**

**Explanation:** The following values are valid for the reason code:

- |           |   |
|-----------|---|
| <b>0</b>  | The CD table for this source has RRN column (iSeries only). The RRN column must be the last column in the table, so the source cannot be altered. |
| <b>1</b>  | The source is a view, and view registrations cannot be altered.   |
| <b>2</b>  | The source is registered for full refresh and cannot be altered.  |
| <b>3</b>  | The source table column does not match the column being altered.  |
| <b>4</b>  | The column is a LOB, DATALINK, or ROWID data type and does not qualify for a before-image value.  |
| <b>5</b>  | The before-image column value cannot be null.   |
| <b>6</b>  | An after-image value has not been registered for the given column.  |
| <b>7</b>  | The before-image prefix cannot be updated if it is used with an existing registered source.   |
| <b>8</b>  | The use of the current before-image prefix makes one of the columns ambiguous in this registered source.  |
| <b>9</b>  | The before-image prefix can be only one character.  |
| <b>10</b> | An internal error occurred.   |

**User Response:** This message is for your information only, and no action is required.

---

**ASN1715E    The Replication action ended in error. The native OS/400 message is *as400native\_message*.**

**Explanation:** An error was encountered while issuing the appropriate command on the OS/400 operating system or iSeries servers. No script is generated.

**User Response:** Refer to the OS/400 Console Log for more detailed error information.

---

**ASN1716W    The Replication action ended with a warning. The native OS/400 message is *as400native\_message*.**

**Explanation:** A warning was encountered while issuing the appropriate command on the OS/400 operating system or iSeries server. A script is generated.

**User Response:** Refer to the iSeries Console Log for more detailed warning information.

---

**ASN1717I    The Replication action ended with an Informational Clause. The native OS/400 message is *as400native\_message*.**

**Explanation:** An informational message was encountered while issuing the appropriate command on the OS/400 operating system or iSeries server. A script is generated.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1718E    The nickname *nicknameowner.nickname* cannot be registered. Reason code *reasoncode*.**

**Explanation:** The nickname is not supported by the replication capture mechanism, as defined. No script is generated. The following values are valid for the reason code:

- |   |  |
|---|--|
| 0 | The internal CCD table (your cd table) already exist.                    |
| 1 | The nickname is on the native catalog table.                             |
| 2 | The nickname is already registered.                                      |
| 3 | A federated registration expects a nickname as a source to be registered |
| 4 | No columns are eligible for the Capture program.                         |
| 5 | The provided nickname is a duplicate                                     |

from an earlier registration, but the corresponding script has not been executed.

- |   |  |
|---|--|
| 6 | A federated registration supports only user tables.  |
| 7 | A federated registration supports only noncondensed and noncomplete CCD tables.  |
| 8 | The CCD nickname provided is a duplicate of a CCD nickname from an earlier registration, but the script for that registration has not yet been executed. |

**User Response:** Check the reason code to determine why the nickname cannot be registered. See the online help for additional explanations or restrictions.

---

**ASN1719W    The non-IBM triggers that were defined for registered source *nicknameowner.nickname* will be dropped. Any additional logic later provided by users in these triggers will be lost.**

**Explanation:** Dropping a registered source implies dropping all the objects that were created during the source registration, regardless of later updates. A script is generated.

**User Response:** Copy the trigger logic before dropping the registered source, if needed.

---

**ASN1720E    Change Data table information for the source nickname *nicknameowner.nickname* is not found in the *capschema.IBMSNAP\_REGISTER* table.**

**Explanation:** A row is found in the *captureschema.IBMSNAP\_REGISTER* table for the given source nickname but the CCD table information for that source is missing. The Change data table information is required to drop the replication definitions. A script is not generated.

**User Response:** Please make sure the correct

source name is given and call the action again.

---

**ASN1722W** The view *view\_owner.viewname* will be registered as full refresh, because all the base tables of this view are registered as full refresh.

**Explanation:** The view must be registered as full refresh, because the base tables of this view are registered as full refresh only or are not registered replication sources.

**User Response:** No action is required.

---

**ASN1723W** The view *viewowner.viewname* will be registered for change-capture replication, because one or more base tables from this view are registered for change-capture replication.

**Explanation:** The view must be registered for change-capture replication, because the base tables of this view are registered for change-capture replication.

**User Response:** No action is required.

---

**ASN1724E** The name of the object that you are creating on the non-DB2 relational server is identical to the *objectowner.objectname* of type *objecttype*.

**Explanation:** The object that you specified cannot be created, because there is an existing object with the same type and same name on the non-DB2 relational server.

**User Response:** Provide a unique name for the object, and reissue the Replication task.

---

**ASN1725W** The trigger named *triggerowner.trigger\_name* already exists on the remote table *remoteowner.remotetablename*. You must not run the generated script until you have determined how to merge the contents of the existing trigger with the generated trigger definition.

**Explanation:** An trigger with this name already exists on the remote table in the non-DB2 relational database. The RDBMS might not indicate a conflict and might subsequently overwrite your existing trigger if you run the CREATE TRIGGER statement in the generated script. Or the RDBMS might return a SQL error indicating that the object already exists. Generated trigger names cannot be customized, because customized triggers cannot be dropped when the registration is dropped.

**User Response:** First, determine how to merge the pre-existing triggers with the generated triggers. Then, either create your own script to merge your existing logic with the trigger logic that is generated by the replication tool, or update the script that is generated by the replication tool to include your existing trigger definitions.

---

**ASN1726W** The trigger named *triggerowner.trigname* does not exist in the remote table *owner.tablename* on the remote server *rmtservername*.

**Explanation:** The trigger does not exist on the remote database. The trigger might have been dropped.

**User Response:** No action is required.

---

**ASN1727I** The registered source *registered\_source* is deactivated.

**Explanation:** The specified registered source has already been deactivated.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1728W The CCSID**  
*Unicode\_ASCII\_EBCDIC of the change data (CD) table cdowner.cdname for the source table sourceowner.sourcetable does not match the CCSID Unicode\_ASCII\_EBCDIC of the IBMSNAP\_UOW table for the capture schema capture\_schema.*

**Explanation:** For the given capture schema, the Apply program will join the IBMSNAP\_UOW table and the CD table of the given source if the column JOIN\_UOW\_CD in the ASN.IBMNSNAP\_MEMBR table is set to Y. This column contains Y if the target type of the associated subscription-set member is not user copy, or if any columns of the IBMSNAP\_UOW table are used in the WHERE clause of the subscription-set member. If the Apply program joins tables with different encoding schemes, an error will occur. For more information about encoding schemes, refer to Appendix B of the Replication Guide and Reference.

**User Response:** For subscription members that will use this registration, define the target table with a type of user copy and do not use any IBMSNAP\_UOW columns in the WHERE clause.

---

**ASN1729E The registration for the nickname nicknameowner.nickname cannot be dropped. The reason code is reasoncode.**

**Explanation:** The registration for this nickname cannot be dropped. No script is generated. The following value is valid for the reason code:

**0** The specified nickname is a duplicate of a nickname included with a prior registration drop. However, the script for that registration drop has not yet been executed.

**User Response:** Review the reason code explanation, and refer to the DB2 Replication Guide and Reference for additional explanations and restrictions.

---

**ASN1800E The subscription set set\_name already exists for the Apply qualifier apply\_qual, Who's On First whos\_onfirst, at the Apply control server server\_alias.**

**Explanation:** There can only be one subscription set with the same name, for a given Apply qualifier and Apply control server. No script is generated.

**User Response:** Create a new set name, or add new members to the existing set.

---

**ASN1801E The statement number statement\_number is associated with a statement string length statement\_stringlength that exceeds the maximum statement length for the Apply qualifier apply\_qual, the set name set\_name, who's on first value whos\_onfirst, at the Apply control server server\_alias.**

**Explanation:** The length of the statement exceeds the allowed limit (1024 in V8). No script is generated.

**User Response:** Rework the statement string so that its length is less than the allowed limits.

---

**ASN1802W The Replication subscription source member is defined with RECAPTURE='N'. All changes to the replica target will not be propagated to the other replica targets.**

**Explanation:** In an update-anywhere scenario, changes made in one target replica will not be recaptured at the source when RECAPTURE='N'. If there is more than one target replica that subscribes to the same source, then the changes made to one target replica will not be reflected in the other replica targets.

**User Response:** If you want the changes propagated to the other replica targets, set RECAPTURE='Y'.

---

**ASN1803I**    **There exist *orphan\_statements* Replication subscription set statements from a previously defined subscription set that was later dropped. These orphan statements are not dropped for the subscription set, for the provided Apply qualifier, at the provided Apply control server.**

**Explanation:** A previous subscription set was dropped, without dropping all its appropriate statements. A script is generated for the new subscription set which shares the same name as the previous subscription set that was dropped. The previous subscription Statements are not dropped.

**User Response:** Issue a Drop Subscription Statements to delete the orphan statements.

---

**ASN1804I**    **The Replication subscription set *MAX\_SYNCH\_MINUTES maxsynch\_minutes* is not within the allowed range for the provided subscription set and Apply qualifier, at the provided Apply control server. The Replication default value is used instead.**

**Explanation:** The valid range for this column is 0 to 999.

**User Response:** No action is required if the default value of 30 minutes is acceptable.

---

**ASN1805I**    **The Replication subscription set *COMMIT\_COUNT commitcount\_value* is not within the allowed range for the provided subscription set and Apply qualifier, at the provided Apply control server. The Replication default value is used instead.**

**Explanation:** The valid range for this column is 0 to 999.

**User Response:** No action is required if the default value of 0 minutes is acceptable.

---

**ASN1806E**    **The replication action ended in error for the Apply qualifier *apply\_qualifier*, subscription-set name *set\_name*, who's on first value *whos\_on\_first*, source member *sourceowner.sourcetable*, source view qualifier *source\_view\_qual*, target member *targetowner.targettable*. The subscription-set member cannot be added to the provided subscription set. Reason code *reason\_code*.**

**Explanation:** The subscription set would be not be valid if the member were added. No script is generated. The following values are valid for the reason code:

- |          |   |
|----------|---|
| <b>0</b> | The subscription set has reached its maximum limit for members.   |
| <b>1</b> | The source member for the Capture schema is not the same as the subscription set for the Capture schema.  |
| <b>2</b> | The iSeries source member is not the same as the subscription set journal.  |
| <b>3</b> | The condensed table member structure is incompatible with the other member structures.  |
| <b>4</b> | The source member does not support change-capture replication, but the target member relies on change-capture. The target structure is either a CCD or replica table, but the source has no CD table. |
| <b>5</b> | The source member is not a complete table.  |
| <b>6</b> | The target member definition expects the existence of the target table, but the target table does not exist.  |
| <b>7</b> | The target member definition asks for creation of the target table, but the target table already exists.  |
| <b>8</b> | The set contains only full refresh  |

	supported target tables, but the new member supports change-capture replication.	25	A replica table with remote journaling is not allowed.
10	The set contains only target tables supported by change-capture replication, but the new member supports full refresh only.	26	An internal CCD table already exists for the specified registered source table.
11	Replica rule: if target member is a replica, the source member can be either a replica or a user table.	27	The source and target servers must be the same for internal CCD tables.
12	The target structure is not supported for this operating system.	28	The internal CCD table must be noncomplete.
13	The target structure is a CCD, which is set as a registration source (autoregistration), but the structure is not complete	29	The source table is remotely journalled and contains LOBs or DATALINK columns.
14	The source member is not registered.	30	No related information exists in the IBMSNAP_PRUNCNTL table.
15	The source member columns have column definitions, but the target type is not an aggregate.	31	No related information exists in the IBMSNAP_PRUNE_SET table.
16	At least one of the excluded target columns from the subscription set is neither nullable nor NOT NULL with defaults.	32	An internal CCD table with a view as a source is not allowed.
17	The target member is a view that cannot be updated.	<b>User Response:</b> Based on the reason code, either create the new member in a different subscription set or create a new subscription set for the new member.	
18	The subscription-set member already exists.	<hr/> <b>ASN1807I    The replication subscription member is added to the provided subscription set and Apply qualifier, at the provided Apply control server with an informational clause. Reason code reason_code.</b>	
19	Unable to find a target column or expression with a valid mapping to the registered source.	<b>Explanation:</b> This message is for your information only, and no action is required. A script is generated. The following values are valid for reason code:	
20	Multiple effective sources have been found but have not been defined consistently.	0	The new set results in a mixture of replica and read-only target members.
21	The external CCD table is noncondensed and contains either DataLink or LOB columns.	1	The subscription set supports transaction commit counts, but the target member does not qualify for transaction processing.
22	The source member journal library or journal name does not match.	2	At least one member has a target member that is a CCD table, but not all members have a target member CCD
23	The remote journal name is not valid.		
24	The journal name or library is not valid.		

table. Different generations of tables are contained in the same set.

- 3 The target member is a non-condensed, non-complete CCD table, without extra columns from IBMSNAP. This target table is essentially the same as the CD table.

**User Response:** Review the reason codes in the explanation, and respond with the following options:

- 0 Consider keeping all the replica tables in one set, and the read-only tables in another.
- 1 Consider keeping all the target types that support transaction processing in the same set, and other tables in another set.
- 2 To maintain a consistent age of data across sets, consider keeping all the CCD target tables, which are part of the same generation, in the same set. Also, consider keeping all non-CCD target tables in a different set. The second set depends on the data being in the first set, as, for example in the middle-tier stage.
- 3 Consider whether you need the CCD target table.

---

**ASN1808E** The Replication action ended in error for Apply qualifier *apply\_qualifier*, set name *set\_name*, Who's On First *whos\_on\_first*, source member *sourceowner.sourcetable*, source view qualifier *source\_view\_qual*, target member *targetowner.targettable*. The subscription target member expects its index key columns to be updated but at least one index key does not have its before-image column registered in the subscription source member.

**Explanation:** The target table allows its index keys to be updated (PRIMARY\_KEY\_CHG = 'Y'). To support this requirement, the Apply program

needs to access the before-image columns of the index keys. So these before-image columns must exist in the Change Data table for the source member. If they do not exist, the Apply program fails. A script is not generated.

**User Response:** For each column of the subscription target index:

- Check if the before-image column for that column is already registered in the source member, at the Capture server.
- If not, register the appropriate before-image column.

---

**ASN1809W** The Replication action expects a subscription index key columns to be updated but the subscription member will be added to the subscription set without allowing updates to the target index key columns. Reason code *reason\_code*.

**Explanation:** In the cases listed above, the PRIMARY\_KEY\_CHG settings are meaningless. A script is generated that was updated with Replication definitions that override user-provided definitions. The following values are valid for reason code:

- 0 The target table type CCD: PRIMARY\_KEY\_CHG is not valid.
- 1 The value of the CHG\_UPD\_TO\_DEL\_INS in the IBMSNAP\_REGISTER table is set to 'Y'.
- 2 The target table is not condensed.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1810W** The Replication subscription source member is defined on the source server with DB2 Referential Integrity constraints, but the subscription target member is a Replica that does not preserve these constraints.

**Explanation:** Referential integrity constraints at the target table are not enforced by DB2 at the replica site. This might not be the intended

behavior at the replica site. A script is generated, and the script might not be ready to execute.

**User Response:** Update the generated script to include the appropriate referential constraints at the target, if needed.

---

**ASN1811W The index definition for the target subscription member cannot guarantee proper uniqueness at the target. Reason code *reason\_code*.**

**Explanation:** The Apply program relies on the unique index definition to correctly update and delete rows in the target table, for some target types. If the provided index does not guarantee uniqueness, the Apply program will have some rework. Understand your application to ensure that this will not be the case. The following values are valid for reason code:

- 0** At least one column is generated by an SQL function, which does not guarantee the uniqueness of the index.
- 1** In a unique index, nullable columns are not generated by SQL functions.

**User Response:** For each column of the subscription target index:

- Check if the column type in the Apply control server ASN.IBMSNAP\_SUBS\_COLS, COL\_TYPE, is 'F'
- If so, redefine the index column expression not to include a SQL expression, or remove that column from the index key (ASN.IBMSNAP\_SUBS\_COLS, IS\_KEY column is set to 'N').

---

**ASN1812E The Replication action ended in error for Apply qualifier *apply\_qualifier*, set name *set\_name*, who's on first value *whos\_on\_first*, source member *sourceowner.sourcetable*, source view qualifier *source\_view\_qual*, target member *targetowner.targettable*. The subscription target member cannot be added because the required target key is not valid. Reason code *reason\_code*.**

**Explanation:** Target members that require a unique index are target types of point-in-time, user copy, and replica tables, and condensed CCDs. If these targets do not have a unique index, the Apply program fails. A script is not generated. The following values are valid for reason code:

- 0** The target table does not already exist but the target key information could not be derived from the source table.
- 1** The target key information cannot be found nor derived, and the RRN is not defined for the CD table (iSeries only).
- 2** The target table or view already exists but the required target key information is missing.
- 3** The target table or view already exists but the target key information is incompatible with the existing partitioning key information.

**User Response:** Define a valid target key. Take the specific actions for the following reason codes:

- 0** Create the appropriate unique index on the source table so that it can be used to derive the replication suggested index.
- 2** Provide the required target key information.
- 3** Refer to the SQL Reference for the DB2 rules on partitioning indexes. For example, the key that you provided might not include the required partitioning key.



---

**ASN1813I**    **The Replication subscription source member is defined on the source server with some DB2 constraints, but the subscription target member does not preserve these constraints. Reason code *reason\_code*.**

**Explanation:** Constraints at the source table are not be enforced by DB2 if they are not specified during the target member definition. This might not be the intended behavior at the replica table server. A script is generated that might not be ready to execute. Constraints are described in the following valid values for reason code:

- 0**            At least one NOT NULL WITH DEFAULT clause in the target member.
- 1**            Partitioned table space.

**User Response:** Update the generated script to include the appropriate DB2 constraints at the target, if needed.

---

**ASN1814E**    **The target column *column\_name* of data type *datatype* cannot be added to subscription target member *tableowner.tablename*. Reason Code *reason\_code*.**

**Explanation:** The subscription member fails the subscription column checks. A script is not generated. The following values are valid for reason code:

- 0**            The column data type is not supported by Replication. Data types that are not supported by DB2 issue message ASN1648E.
- 1**            The target data type is incompatible with the corresponding source data type.
- 2**            The column is not found in the source table registration.
- 3**            The column type is not supported for federated targets.
- 4**            The target column is a LOB. The maximum number of LOB columns is exceeded for the target member.

- 5**            The source column contains SQL column function, but the target member structure is neither base aggregate nor change aggregate.
- 6**            The target table type is replica, and the source column is a LOB column.
- 7**            The target table type is replica, and source column is DATALINK value. But the CONFLICT\_LEVEL > 0.
- 8**            A noncondensed CCD target table with LOB columns is not supported.
- 9**            The column is not in the existing target table.

**User Response:** Review the reason code in the explanation and respond as follows:

- 0**            Change the data type to one that is supported.
- 1**            Make sure the target data type matches the source data type.
- 2**            Register the column of the source table.
- 3**            Choose a valid data type that is supported for federated targets.
- 4**            Make sure the number of LOB columns at the target member does not exceed the allowable limit.
- 5**            Change either the source column expression or the target table structure.
- 6**            Remove the LOB column for the replica target from the subscription member.
- 7**            Remove the DATALINK column from the subscription member if the replica needs a conflict level greater than 0. Otherwise, change the replica conflict level.
- 8**            Remove the LOB columns.
- 9**            Verify the column name.

---

**ASN1815E** The Replication action ended in error. The subscription set *set\_name* for Apply qualifier *apply\_qual*, Who's On First *whos\_on\_first* is to be dropped if empty, but at least one member exists for this set. The subscription set cannot be dropped.

**Explanation:** The subscription set is not dropped because at least one member exists in the ASN.IBMSNAP\_SUBS\_MEMBER at the provided Apply control server, for the provided Apply qualifier of the particular subscription set. A script is not generated.

**User Response:** Drop the subscription members that still exist, and then drop the subscription set. Alternatively, issue the *Drop Subscription Set* task, with no requirement for the subscription set to be empty.

---

**ASN1816W** The Replication subscription set contains at least one member that will be dropped once the subscription set is dropped.

**Explanation:** When a subscription set is dropped successfully, all the set members are automatically dropped also.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1818W** The Replication subscription member is updated with new subscription Where Clause predicates. Previous predicates already exist for the subscription member. They will be overwritten by the new predicate information.

**Explanation:** The specified member already contains a predicate. The new predicate overwrites the old one. A script is generated.

**User Response:** Provide the complete predicate clause for the replication task. You might need to update the predicate clause if does not contain all of the existing predicate clause.

---

**ASN1819W** The Replication subscription set is disabled successfully. Note however that disabling a subscription set has a direct impact to the Capture pruning logic of all source members for that subscription set.

**Explanation:** The Capture pruning logic does not prune any CD table until the dependent subscription members have been populated by the Apply program. A script is generated. It might need to be updated, if the disabling of the subscription set is not the ideal choice, and dropping the subscription set is a better option. See below.

**User Response:** If the subscription set is going to be disabled for a considerable amount of time that the pruning process of the CD tables will be impacted, or if the impact to the CD tables for the dependent registered sources will impact dramatically the Capture program and the Capture server CD tables, then consider dropping the subscription set and creating it again later, instead of simply disabling it. Alternatively, deactivate the appropriate registrations.

---

**ASN1820E** The Replication string for subscription set *set\_name*, apply qualifier *apply\_qual*, Who's On First *whos\_on\_first* contains DB2 syntax that is not valid. The string type is *string\_type*, string text is *string\_text* and SQL Message is *sql\_message*.

**Explanation:** The specified string is not a valid. A script is not generated.

**User Response:** Please correct the appropriate object syntax and issue the Replication task again.

---

**ASN1821W** Dependent subscription sets will no longer be valid after the existing subscription set is dropped, if this subscription set contains target members that are registered sources at its target server.

**Explanation:** The dependent subscriptions rely on their source member tables to exist. If these source members are maintained as replication targets, and these targets are dropped, then the Apply program fails when it processes the dependent subscription sets. Dependent subscription sets might be impacted if the captureserver.IBMSNAP\_PRUNCNTL table at the target server contains SOURCE\_OWNER or SOURCE\_TABLE rows for which these values are the target tables being dropped. A script is generated.

**User Response:** Deactivate or drop the dependent subscription sets before running the script, if required.

---

**ASN1822E** The Replication action ended in error for Apply qualifier *apply\_qual*, set name *set\_name*, source member *sourceowner.sourcename*, target member *targetowner.targetname*. The provided subscription member does not exist for the provided subscription set.

**Explanation:** The specified member cannot be found in the ASN.IBMSNAP\_SUBS\_MEMBR for the provided Apply qualifier at the provided Apply control server.

**User Response:** Make sure the Apply qualifier, set name, member name, and control server provided are correct.

---

**ASN1823E** The subscription set *set\_name* does not exist for the Apply qualifier *apply\_qual*, Who's On First *whos\_onfirst*, at the Apply control server *server\_alias*.

**Explanation:** The specified subscription set cannot be found in the ASN.IBMSNAP\_SUBS\_SET for the provided Apply qualifier at the provided Apply control server.

**User Response:** Make sure the Apply qualifier, set name, member name, and control server provided are correct.

---

**ASN1824W** The Replication subscription set was updated with a COMMIT\_COUNT of 0.

**Explanation:** The source is a view of multiple tables, and the commit count is null for the set. A Commit count of 0 is enforced for the set.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1825W** The Replication action to drop a member did not drop the view.

**Explanation:** Even though the request was made to drop the view the action was not completed as requested.

**User Response:** You need to manually drop the view.

---

**ASN1826W** The Capture schema *capture\_schema* does not exist at the provided source server.

**Explanation:** Please make sure the Capture schema exists before adding any members to the subscription set.

**User Response:** Create the Capture server control tables at the source server with the Capture schema specified.

---

**ASN1827W** The column *target\_columnname* of the target member *target\_member* does not preserve a DB2 column attribute of the corresponding column *source\_columnname* of the source member *source\_member*. Reason Code *reason\_code*.

**Explanation:** A DB2 column attribute of the source column differs from the corresponding target column. The following values are valid for reason code:

1

The source column is nullable and the target column is not nullable.

2

The source column is not nullable and the target column is nullable.

3

The source column has a default value and the target column has none.

4

The target column has a default value and the source column has none.

**User Response:** If the reason code is 1, then check whether there are null values in the source column that will be applied to the target column. If necessary, change the target column to NULLABLE. Alternatively, update the generated script to include the appropriate DB2 attributes at the target, if needed.

---

**ASN1828E** The Replication action ended in error for Apply qualifier *apply\_qual*, set name *set\_name*, source member *sourceowner.sourcename*, target member *targetowner.targetname*. When the subscription target server is a non-IBM target server, either the action is not supported or is supported with restrictions. Reason Code is *reason\_code*.

**Explanation:** These are current restrictions. No script is generated. The following values are valid for reason code:

0 Not supported.

1 Supported for the following target table structures: point-in-time, CCD, user copy.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1829I** A valid nickname *nicknameowner.nickname* is found for the subscription target table. Column data type mapping rules are enforced. The nickname is used as is.

**Explanation:** An existing target nickname was found in the federated database that is valid for this subscription (the column data type mapping checks are valid); however, there is no check to validate the existence of the target table on the non-DB2 relational database server.

**User Response:** Make sure that the remote table exists for the provided nickname. Otherwise, the Apply program fails.

---

**ASN1830E**    **The Replication action ended in error for the Apply qualifier *apply\_qualifier*, subscription set *set\_name*, whos on first value of *whos\_on\_first*, source member *sourceowner.sourcetable*, source view qualifier *source\_view\_qual*, target member *targetowner.targettable*, and predicate of *predicate*. The Subscription-set member cannot be added to this subscription set. The reason code is *reason\_code*.**

**Explanation:** The subscription-set member is invalid, and no script is generated. A possible reason code is:

**0**            The predicate references columns from non-existing CD or UOW tables.

**User Response:** Verify the accuracy of the specified predicate, and refer to the documentation on advanced change predicate features.

---

**ASN1831E**    **The Replication action ended in error. No subscription statements exist for the subscription set *set\_name* for the Apply qualifier *apply\_qual*, with a whos on first value of *whos\_onfirst*, at the Apply control server *control\_server*.**

**Explanation:** No subscription statements exist for the specified subscription-set name with this Apply qualifier.

**User Response:** Verify that the specified subscription-set name under this Apply qualifier contains subscription-set statements.

---

**ASN1832W**    **A column named *column\_name* already exists in the ASN.IBMSNAP\_SUBS\_COLS control table.**

**Explanation:** The specified column already exists in the ASN.IBMSNAP\_SUBS\_COLS table.

**User Response:** No action is required.

---

**ASN1833E**    **The CCSID *Unicode\_ASCII\_EBCDIC* of the change data (CD) table *cdowner.cdname* for the source table *sourceowner.sourcetable* does not match the CCSID *Unicode\_ASCII\_EBCDIC* of the IBMSNAP\_UOW table for the capture schema *capture\_schema*. The provided subscription member definition would require a join of these two tables.**

**Explanation:** For the given capture schema, the Apply program will join the IBMSNAP\_UOW table and the CD table of the given source if either the target type of the associated subscription-set member is not user copy, or if any columns of the IBMSNAP\_UOW table are used in the WHERE clause of the subscription-set member. If the Apply program processes such a subscription-set member defined with the given source table and capture schema by joining the CD table of the source table with the IBMSNAP\_UOW table, an error will occur because of the different encoding schemes of the tables. For more information about encoding schemes, refer to Appendix B of the Replication Guide and Reference.

**User Response:** You can either

- select a target type of user copy and do not use columns of the IBMSNAP\_UOW table in the WHERE clause of the subscription member, or
- register the source using a different capture schema and create the CD table in a tablespace with the same encoding scheme as those of the IBMSNAP\_UOW table of the new capture schema.

---

**ASN1834W**    **The default target capture schema of 'ASN' will be used for the subscription set.**

**Explanation:** The subscription set requires a target Capture schema, and the default value of 'ASN' is used.

**User Response:** No action is required if the

default is appropriate for the target Capture Schema column in this subscription set.

---

**ASN1835W** The target column *column\_name* of data type *datatype* has been added to the subscription-set member target *tableowner.tablename*, but the corresponding source column *column\_name* of data type *datatype* can contain data that is not applicable to the target column.  
**Reason code** *reason\_code*.

**Explanation:** The source column definition does not exactly match with the target column definition. Therefore, if the data that is selected from the source by the Apply program is not appropriate for the target column, the Apply program might fail or might modify the source data (by truncating it). Note: If your application does not generate data that will make the Apply program fail, there is no problem with the definition mismatch.

The following values are valid for the reason code:

1

The target column length is less than the resolved source column expression.

2

The target column scale is less than the resolved source column expression.

3

The target column precision is less than the resolved source column expression.

4

The target and source column data types are compatible only for certain source values.

**User Response:** If possible, change the definitions at the target site to be compatible with the source definitions. (This is usually driven by the applications that run at the target site.)

If you must keep the definition mismatch (because you have a specific reason why the

target definitions must be different than the source definitions), review your applications to ensure that the definition mismatch will not cause a run-time problem.

---

**ASN1836W** The target table *owner.name* will not be dropped because it is registered as a source under the Capture schema *schemaname*.

**Explanation:** The target table is registered as a source under the specified Capture schema. If the table is dropped, the registration will no longer be valid.

**User Response:** Drop the registration for the table, and then drop the table.

---

**ASN1837W** The DB2 target *tableowner.tablename* is not dropped.

**Explanation:** The target table is a replica or an external Consistent Change Data table (CCD) and it is also registered at the target server so it may be the source for dependent targets. The table cannot be dropped automatically.

**User Response:** Drop the registered source for the replica or external CCD table. Then, manually drop the DB2 target table or delete the replica or external CCD subscription member. The deletion of the member will drop the DB2 target table.

---

**ASN1900E** The table or view *objectowner.objectname* cannot be promoted to the new server.  
**Reason code** *reason\_code*.

**Explanation:** The following values are valid for reason code:

- 0 The table type on this operating system is not supported for promote request.
- 1 The source server operating system required for the promote needs to match the target server operating system.
- 2 The table or view does not exist.

**User Response:** Review the reason code in the explanation and respond as follows:

- 0 This message is for your information only, and no action is required.
- 1 Current restriction.
- 2 Verify that the table or view exists on the source server operating system.

---

**ASN1901E** The registered source *sourceowner.sourcename* cannot be promoted for Capture schema *captureschema* at Capture server *capture\_server*. Reason code *reason\_code*.

**Explanation:** The following values are valid for the reason code:

- 0 The table or view is not a registered source.
- 1 The registered source is a replica table.
- 2 The registered source is on DB2 for iSeries but has a remote journal.
- 3 The table or view has already been promoted.
- 4 A view on a view is not supported by the replication promote registration function.

**User Response:** Review the reason code in the explanation, and respond as follows:

- 0 The table or view name that you specified in the IBMSNAP\_REGISTER table contained no entries for the specified Capture schema. The table or view registration cannot be promoted for this particular Capture schema.
- 1 The table you specified is of type replica (with a SOURCE\_STRUCTURE column value of 7) in the *captureschema.IBMSNAP\_REGISTER* table. The table cannot be promoted as a registered source. A replica can be promoted only within the context of a subscription set to ensure that proper definitions are maintained between the source user table and the replica target.
- 2 The registered source is maintained on

DB2 for iSeries with a remote journal, which can be promoted only with SQL script.

---

**ASN1902W** Make sure that the schemas exist on the promoted Capture server before running the script. The Replication definitions will be incomplete if the object does not exist at the promoted Capture server.

**Explanation:** The promote tasks allows you to provide a new Capture server and new Capture server schemas. However, the promote tasks do not connect to the new Capture server to verify the names and existence of the Capture server and schemas. You must verify this information before running the script to ensure that the script executes successfully.

**User Response:** To generate the required objects, run the appropriate SQL prior to running the script.

---

**ASN1903W** The object *objectowner.objectname* does not exist on the promoted Apply control server, yet some promoted objects depend on its existence. Failure to create this object will result in incomplete Replication definitions at promoted Apply control server. Reason code *reason\_code*.

**Explanation:** Since the promote tasks allow you to provide a new Apply control server name, the tasks detect whether some required objects exist to ensure proper execution of the generated script. A script is generated, but is not ready to run. The following values are valid for reason code:

- 0 The Apply control server control tables do not exist.
- 1 The registration information for all source members of a promoted set.

**User Response:** To generate the required objects, run the appropriate SQL prior to running the script.

---

**ASN1904I**    **The Replication subscription member is promoted successfully for the provided Apply qualifier, at the provided Apply control server. Reason code *reason\_code*.**

**Explanation:** This message is for your information only; no action is required. A script is generated that might need some updates before being executed. The following values are valid for reason code:

- 0**            The source member structure is incompatible with the target member structure.
- 1**            The target member is a replica (replica1) that is also the source member of another replica (replica2.) The RECAPTURE value for the registration row of replica2 does not allow updates from the user table to be replicated at replica2.
- 2**            The source member is a user table that is also the source member of more than one replica (replica1 and replica2). The RECAPTURE value for the registration row of the user table does not allow updates at replica1 to be replicated at replica2, and vice-versa.

**User Response:** Review the reason code in the explanation, and respond as follows:

- 0**            Check the ASN.IBMSNAP\_SUBS\_MEMBR table, TARGET\_STRUCTURE column. The value in the column should be compatible with the corresponding source member captureschema.IBMSNAP\_REGISTER table, SOURCE\_STRUCTURE column.
- 1, 2**        Update the values, if needed.

---

**ASN1905W**    **The Capture server alias and the Capture schema name on both the host system and new system are the same. The generated replication definitions cannot work if run on the host system.**

**Explanation:** The promote task detected that the Capture server alias and the Capture schema name are the same on both on the host and new system. The generated SQL script must be modified, or it will fail when executed.

**User Response:** Take one of the following actions: 1) Run the same task with different Capture server alias and Capture schema name values for the host or new system. 2) Or, change the Capture server alias or Capture schema name in the generated script.

---

**ASN1950E**    **ASNCLP : An unexpected token *token\_name* was found. Valid tokens include *list\_of\_tokens*.**

**Explanation:** The command was entered with incorrect syntax.

**User Response:** Check the documentation to verify your command syntax.

---

**ASN1951E**    **ASNCLP : The command was entered with profile *profile\_name* that is not valid.**

**Explanation:** A profile must exist before it can be used in a command.

**User Response:** Issue the corresponding SET PROFILE command, and then re-enter the command that failed.

---

**ASN1952E**    **ASNCLP : The program encountered an internal error.**

**Explanation:** The Replication command line processor encountered an unrecoverable error condition.

**User Response:** Obtain the log file with the error, and contact IBM Software Support.



---

**ASN1953I ASNCLP : Command completed.**

**Explanation:** All commands of this ASNCLP session completed successfully. Please note that some individual commands in this session may have produced errors, warnings, or informational messages.

**User Response:** Check the ASNCLP log file for any errors, warnings, or informational messages produced by the commands in this session.

---

**ASN1954E ASNCLP : Command failed.**

**Explanation:** At least one command in the ASNCLP session failed, and processing stopped.

**User Response:** Look in the ASNCLP log file to diagnosis the error. Then fix the error, and try the command again.

---

**ASN1955I ASNCLP : The program will use the following files:**  
*capture\_script\_file\_name* for the **Capture SQL script**,  
*control\_script\_file\_name* for the **control SQL script**,  
*target\_script\_file\_name* for the **target SQL script**, and *log\_file\_name* for the **log file**.

**Explanation:** The ASNCLP session generated information in the specified files.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1956I ASNCLP : The program now generates the script for action:**  
*action\_name*.

**Explanation:** All input for this command has been successfully parsed, and the command that generates the script is now invoked.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1957E ASNCLP : The value *value* for the input parameter *input\_parameter* is incorrect. The reason code is *reason\_code*.**

**Explanation:** The value of the input parameter is incorrect. The following values are valid reason codes:

- 1** The input parameter is a character value but should be a numeric value.
- 2** The input parameter is a numeric value but should be a character value.
- 3** The command line processor cannot access the specified file.

**User Response:** Check the reason code, and provide a valid input parameter value.

---

**ASN1976E *pgmname* : *program\_qualifier*. The specified database alias *db\_alias\_name* already exists in the password file *password\_file\_name*.**

**Explanation:** The key that you specified already exists in the password file.

**User Response:** Enter this command again using the MODIFY parameter instead of the ADD parameter.

---

**ASN1977E *pgmname* : *program\_qualifier*. The value of the input parameter *parameter\_name* is missing.**

**Explanation:** The above input parameter must be specified.

**User Response:** Invoke the utility again using a valid input parameter value.

---

**ASN1978E *pgmname* : *program\_qualifier*. The value of the input parameter *parameter\_name* is too long.**

**Explanation:** The asnpwd command supports a maximum of eight characters for the length of the database alias and a maximum of 128 characters for the length of both the user ID and the password. The specific lengths of the user ID and the password are dependent upon the

operating system that you are using.

**User Response:** Invoke the API using an input parameter with a valid length.

---

**ASN1979E** *pgmname : program\_qualifier. The program encountered an unexpected token token\_name. Expected tokens include list\_of\_tokens.*

**Explanation:** The command was entered with incorrect syntax.

**User Response:** Check the documentation to verify your command syntax.

---

**ASN1980E** *pgmname : program\_qualifier. The program did not complete successfully because reason.*

**Explanation:** The asnpwd command encountered system problems as indicated in the message.

**User Response:** Take action based on the information in the message. Enter the command again after fixing the error.

---

**ASN1981I** *pgmname : program\_qualifier. The program completed successfully using password file password\_file\_name.*

**Explanation:** The asnpwd command completed successfully.

**User Response:** This message is for your information only, and no action is required.

---

**ASN1982E** *pgmname : program\_qualifier. The specified database alias db\_alias\_name does not exist in the password file password\_file\_name.*

**Explanation:** The key that you specified with the MODIFY or DELETE parameter does not exist in the password file.

**User Response:** Enter the command again using the ADD parameter.

---

**ASN1983E** *pgmname : program\_qualifier. The program cannot find the password file password\_file\_name.*

**Explanation:** No password file was found.

**User Response:** Verify that the password file exists in the specified path. If you are using the Password Management utility for the first time, use the INIT parameter.

---

**ASN1984E** *pgmname : program\_qualifier. The program cannot be initialized because the password file password\_file\_name already exists.*

**Explanation:** The password file already exists in the specified path.

**User Response:** Verify that the password file has been deleted. Then retry the command.

---

**ASN1985E** *pgmname : program\_qualifier. The program encountered an internal error when using the password file password\_file\_name.*

**Explanation:** The operating system produced an unexpected error when trying to access the password file. No information about this error is available. However, this error might have occurred if the password file was manually edited causing the format of the file to change.

**User Response:** Retry the command. If the problem persists, use the INIT parameter to create a new password file.

---

**ASN5101I** **MONITOR** *monitor\_qualifier. The Replication Alert Monitor program started successfully.*

**Explanation:** This message appears after a successful start of the Replication Alert Monitor program.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5102I**    **MONITOR** *monitor\_qualifier*. The Replication Alert Monitor program initialized successfully and is monitoring *number-of-alert-conditions* alert conditions.

**Explanation:** The Replication Alert Monitor program successfully started.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5103I**    **MONITOR** *monitor\_qualifier*. The Replication Alert Monitor program re-initialized successfully and is monitoring *number-of-alert-conditions* alert conditions.

**Explanation:** The Replication Alert Monitor program successfully re-initialized.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5104W**    **MONITOR** *monitor\_qualifier*. *number-of-alert-conditions* alert conditions were ignored.

**Explanation:** The Replication Alert Monitor program initialized or re-initialized. Some alert conditions that are not valid might be excluded as noted in previously issued messages.

**User Response:** Check the IBMSNAP\_CAPTRACE table for messages about excluded alert conditions.

---

**ASN5107I**    **MONITOR** *monitor\_qualifier*. The Replication Alert Monitor program stopped.

**Explanation:** The Replication Alert Monitor program terminated.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5110I**    **MONITOR** *monitor\_qualifier*. The Alert Monitor program encountered an SQL error. The **ERRCODE** is *error\_code*. The **SQLSTATE** is *sqlstate*. The **SQLCODE** is *sqlcode*. The **SQLERRM** is *sqlerrm*. The **SQLERRP** is *sqlerrp*. The server name is *server\_name*. The table name is *table\_name*.

**Explanation:** An error occurred during the execution of an SQL statement.

**User Response:** Refer to your database message reference for an explanation of the SQL error code.

---

**ASN5111I**    **MONITOR** *monitor\_qualifier*. *number-of-rows* rows were pruned from the table *schema.table-name*.

**Explanation:** The Replication Alert Monitor program pruned rows from the IBMSNAP\_ALERTS or the IBMSNAP\_MONTRACE table.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5112E**    **MONITOR** *monitor\_qualifier*. The pruning program failed with a **SQLCODE** of *sqlcode* when the program pruned the table *schema.table-name*.

**Explanation:** The pruning program failed when executing a pruning operation.

**User Response:** Read the corresponding action regarding this SQLCODE, and correct the error.

---

**ASN5116E**    **MONITOR** *monitor\_qualifier*. The monitor qualifier does not exist on the monitor control server *monitor-server*.

**Explanation:** The Replication Alert Monitor program cannot find the monitor qualifier on this monitor control server.

**User Response:** Verify that the monitor qualifier

name used with the monitor\_qual parameter is correct.

---

**ASN5117E**    **MONITOR** *monitor\_qualifier*. **There are no valid Alert Conditions for this monitor qualifier on Monitor Server** *monitor\_server*.

**Explanation:** The Replication Alert Monitor program cannot find any alert conditions for this monitor qualifier.

**User Response:** Verify that the monitor qualifier name used with the monitor\_qual parameter is correct, and check that the alert conditions are enabled. Also, check any previously issued messages.

---

**ASN5118E**    **MONITOR** *monitor\_qualifier*. **The program cannot connect to the monitor control server** *server\_name*. **The SQLCODE is** *sqlcode*, **and the SQLSTATE is** *sqlstate*.

**Explanation:** The Monitor program tried to connect to the monitor control server and failed with the corresponding SQLCODE.

**User Response:** Read the corresponding action for this SQLCODE, and correct the error.

---

**ASN5119E**    **MONITOR** *monitor\_qualifier*. **The program cannot connect to the server** *server\_name*. **The SQLCODE is** *sqlcode*, **and the SQLSTATE is** *sqlstate*.

**Explanation:** An SQL CONNECT statement failed when the Monitor program tried to connect to the monitored Capture or Apply control server.

**User Response:** Read the corresponding action regarding this SQLCODE, and correct the error.

---

**ASN5121E**    **MONITOR** *monitor\_qualifier*. **A contact does not exist for the alert condition with a component of component, a server of server, a schema or qualifier of schema\_or\_qualifier, and a condition name of condition.**

**Explanation:** The specified contact does not exist in the IBMSNAP\_CONTACTS table for this alert condition.

**User Response:** Verify the contact information, and correct the alert condition.

---

**ASN5122E**    **MONITOR** *monitor\_qualifier*. **The contact group** *group-name* **does not exist or is empty. The component is component, the server is server, the schema or qualifier is schema\_or\_qualifier, and the condition name is condition.**

**Explanation:** The contact group specified in an alert condition does not have corresponding contacts in the IBMSNAP\_CONTACTGRP table or does not exist in the IBMSNAP\_CONTACTGRP table. A contact group cannot be empty.

**User Response:** Verify the contacts for this group, and correct the alert condition.

---

**ASN5123E**    **MONITOR** *monitor\_qualifier*. **The table** *table-name* **is not found. The Capture control server is capture-server. The schema is schema. The condition name is condition-name.**

**Explanation:** The Replication Alert Monitor program cannot find a table when attempting to monitor a condition on the Capture control server.

**User Response:** Verify that the table exists on the Capture control server, or correct the alert condition.

---

**ASN5124E**    **MONITOR** *monitor\_qualifier*. **The table *table-name* is not found. The Apply control server is *apply-control-server*. The Apply qualifier is *apply-qualifier*. The subscription-set name is *set-name*. The condition name is *condition-name*.**

**Explanation:** The Replication Alert Monitor program cannot find a table when attempting to monitor a condition on the Apply control server.

**User Response:** Verify that the table exists on the Apply control server, or correct the alert condition.

---

**ASN5125E**    **MONITOR** *monitor\_qualifier*. **The Apply qualifier *apply-qualifier* or the subscription set *set-name* is not found.**

**Explanation:** The Replication Alert Monitor program cannot find the Apply qualifier or the subscription set when attempting to monitor a condition on the Apply control server.

**User Response:** Verify that the Apply qualifier and the subscription set exist on the Apply control server, or correct the alert condition.

---

**ASN5126E**    **MONITOR** *monitor\_qualifier*. **There is an error sending a notification. The return code is *rc*.**

**Explanation:** When an alert notification was attempted, the ASNMAIL exit returned an error. The reasons for the error may include:

- 1**        SMTP protocol failed. Verify the address of your e-mail server with your administrator.
- 2**        SMTP socket failed. Verify the configuration of your e-mail server or client.
- 3**        The e-mail address is not valid. Verify the e-mail address.
- 4**        Software error.
- 99**       ASNMAIL exit not found.

**User Response:** For return codes 1 and 2, verify the configuration of your e-mail server and client. For return code 3, check whether the e-mail address is correct. For return code 99, verify if the ASNMAIL exit properly installed.

---

**ASN5127E**    **MONITOR** *monitor\_qualifier*. **An invalid value *value* exists in the column *column-name* of the table *table-name*.**

**Explanation:** This message indicates that the Replication Alert Monitor program found a column with a value that is not valid during program initialization.

**User Response:** Verify the values of the column definitions in the specified table.

---

**ASN5129I**    **MONITOR** *monitor\_qualifier*. **The Replication Alert Monitor on server *server-name* reports an e-mail alert.**

**Explanation:** The Replication Alert Monitor program sent an e-mail alert.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5130I**    **MONITOR** *monitor\_qualifier*. ***capture\_message*. The Capture control server is *capture-server*. The schema is *schema*. The monitor control server is *monitor-server*.**

**Explanation:** The Replication Alert Monitor program retrieved a Capture program message from the IBMSNAP\_CAPTRACE table when processing CAPTURE\_ERRORS or CAPTURE\_WARNINGS conditions.

**User Response:** Read the Capture program message, and take appropriate action. Also, check any errors or warnings from the Capture control server.

---

**ASN5131I**    **MONITOR** *monitor\_qualifier*.  
*apply\_message*. **The Apply control  
server is *apply-server*. The Apply  
qualifier is *apply-qualifier*. The  
monitor control server is  
*monitor-server*.**

**Explanation:** The Replication Alert Monitor program retrieved an Apply program message from the IBMSNAP\_APPLYTRAIL or the IBMSNAP\_APPLYTRACE table when processing APPLY\_SUBSFALING, APPLY\_ERRORS, or APPLY\_WARNINGS alert conditions.

**User Response:** Read the Apply program message, and take appropriate action. Also, check any errors or warnings from the Apply control server.

---

**ASN5133I**    **MONITOR** *monitor\_qualifier*. **The  
following alert *message\_number* has  
occurred *number\_of\_times* times in  
the last *number\_of\_minutes*  
minutes. The notification for this  
alert will be suspended.**

**Explanation:** This message is issued after an alert has been detected the number of times specified in the MAX\_NOTIFICATIONS\_PER\_ALERT parameter (the default is 3) for the number of minutes specified in the MAX\_NOTIFICATIONS\_MINUTES parameter (the default is 60 minutes).

**User Response:** This message is for your information only, and no action is required.

---

**ASN5134I**    **MONITOR** *monitor\_qualifier*. **Alerts  
issued.**

**Explanation:** The text of this message appears in the subject line of the e-mail alerts sent by the Replication Alert Monitor program.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5135W**    **MONITOR** *monitor\_qualifier*. **Too  
many alerts *number\_of\_alerts* for  
server *server\_name* between  
*lower\_bound\_time* and  
*upper\_bound\_time*, schema or  
qualifier *schema\_qual\_name*,  
condition name *condition\_name*.**

**Explanation:** The Alert Monitor reached the maximum of alerts allowed for a monitor cycle (1024), or memory can not be allocated for them. The Alert Monitor will send the notifications and will update the Monitor control server and will reconnect to the server starting in the next condition. Some alerts for the specified alert condition might not be sent and inserted in the Monitor control server.

**User Response:** You should verify the specified alert condition directly in the server to check if alerts were lost.

---

**ASN5136W**    **MONITOR** *monitor\_qualifier*. **There  
has been an error calling DAS  
component. The return code is *rc*  
for server *server\_name* for schema  
or qualifier *schema\_qual\_name* and  
condition name *condition\_name*.**

**Explanation:** While processing the condition name for the given server, the DAS component returned an error.

**User Response:** Verify that DAS is running properly in both the client and the remote server.

---

**ASN5150W**    **MONITOR** *monitor\_qualifier*. **The  
Capture program is not running.  
The Capture control server is  
*capture\_server*, and the schema is  
*schema*.**

**Explanation:** The alert condition CAPTURE\_STATUS indicates that the Capture program is not running.

**User Response:** Verify the status of the Capture program on the specified Capture control server.

---

**ASN5151W** **MONITOR** *monitor\_qualifier*. The elapsed time since the last commit of Capture program exceeds the threshold value. The Capture control server is *capture\_server*. The schema is *schema*. The last commit time is *time*. The threshold is *minutes minutes*.

**Explanation:** The CAPTURE\_LASTCOMMIT alert condition detects that the difference between the current timestamp value and the value of the MAX\_COMMIT\_TIME column in the IBMSNAP\_RESTART table is greater than the threshold value of this alert condition as specified by the PARM\_INT column value in the IBMSNAP\_CONDITIONS table.

**User Response:** Check the Capture control server, and determine the reason for the commit delay.

---

**ASN5152W** **MONITOR** *monitor\_qualifier*. The current Capture latency exceeds the threshold value. The Capture control server is *capture\_server*. The schema is *schema*. The Capture latency is *latency seconds*. The threshold is *threshold seconds*.

**Explanation:** The CAPTURE\_CLATENCY alert condition detects that the difference between the CURR\_COMMIT\_TIME and the MAX\_COMMIT\_TIME column values in the IBMSNAP\_RESTART table is greater than the threshold value of this alert condition as specified by the PARM\_INT column value in the IBMSNAP\_CONDITIONS table.

**User Response:** Check the Capture control server, and determine the reason for the Capture latency.

---

**ASN5153W** **MONITOR** *monitor\_qualifier*. The historic Capture latency exceeds the threshold value. The Capture control server is *capture\_server*. The schema is *schema*. The Capture latency is *latency seconds*. The threshold is *threshold seconds*.

**Explanation:** The CAPTURE\_HLATENCY alert condition detects that the difference between the MONITOR\_TIME and the SYNCHTIME column values in the IBMSNAP\_CAPMON table is greater than the threshold value of this alert condition as specified by the PARM\_INT column value in the IBMSNAP\_CONDITIONS table. This latency value might correspond to past latency but might help track latency trends over time.

**User Response:** Check the Capture control server, and determine the reason for the Capture latency.

---

**ASN5154W** **MONITOR** *monitor\_qualifier*. The memory used by the Capture program exceeds the threshold value. The Capture control server is *capture\_server*. The schema is *schema*. The amount of memory used is *memory megabytes*. The threshold is *threshold megabytes*.

**Explanation:** The CAPTURE\_MEMORY alert condition detects that the value of the CURRENT\_MEMORY column in the IBMSNAP\_CAPMON table is greater than the threshold value of this alert condition as specified by the PARM\_INT column value in the IBMSNAP\_CONDITIONS table.

**User Response:** Check the Capture control server, and determine the reason for the excessive memory usage. Modify the memory\_limit parameter of the Capture program, if necessary.

---

**ASN5160W** **MONITOR** *monitor\_qualifier*. The Apply program is not running. The Apply control server is *apply\_server*, and the Apply qualifier is *apply-qualifier*.

**Explanation:** The APPLY\_STATUS alert condition detects that the Apply program is not running.

**User Response:** Verify the status of the Apply program at the specified Apply control server.

---

**ASN5161W** **MONITOR** *monitor\_qualifier*. The subscription set is inactive and is in an error state. The Apply control server is *apply-control-server*. The Apply qualifier is *apply-qualifier*. The name of the subscription set is *set-name*. The who's on first value is *wof*, and the lastsucces value is *timestamp*.

**Explanation:** The APPLY\_SUBSINACT alert condition detects that the subscription set is inactive and that the status is not equal to zero (0).

**User Response:** If the subscription set should be active, check this subscription set on the Apply control server.

---

**ASN5162W** **MONITOR** *monitor\_qualifier*. A full refresh occurred. The Apply control server is *apply-control-server*. The Apply qualifier is *apply-qualifier*. The name of the subscription set is *set-name*. The who's on first value is *wof*, and the target table is *table-name*.

**Explanation:** The APPLY\_FULLREFRESH alert condition detects that the target table was refreshed during the past monitor cycle.

**User Response:** If the full refresh was in error, verify the cause of the full refresh for this specified target table.

---

**ASN5163W** **MONITOR** *monitor\_qualifier*. The subscription is delayed beyond the threshold. The Apply control server is *apply-control-server*. The Apply qualifier is *apply-qualifier*. The name of the subscription set is *set-name*. The who's on first value is *wof*. The time delayed is *time*, and the threshold is *threshold* seconds.

**Explanation:** The APPLY\_SUBSDELAYED alert condition detected a subscription set that complies with the following conditions: CURRENT TIMESTAMP minus LASTRUN is greater than the threshold.

**User Response:** Check any previous messages to see if this subscription set has an error and to verify that the Apply program is running.

---

**ASN5164W** **MONITOR** *monitor\_qualifier*. The rows reworked in a subscription exceeds the threshold. The Apply control server is *apply-control-server*. The Apply qualifier is *apply-qualifier*. The name of the subscription set is *set-name*. The who's on first value is *wof*. The number of reworked rows is *rows*, and the threshold is *threshold* rows.

**Explanation:** The APPLY\_REWORKED alert condition detects a subscription set with a SET\_REWORKED column value (in the IBMSNAP\_APPLYTRAIL table) that exceeds the specified threshold.

**User Response:** Verify the reason why this number of rows have been reworked.



---

**ASN5165W** **MONITOR** *monitor\_qualifier*.  
Transactions have been rejected in the subscription set. The Apply control server is *apply-control-server*. The Apply qualifier is *apply-qualifier*. The name of the subscription set is *set-name*. The who's on first value is *wof*. The number of rejected transactions is *transactions*.

**Explanation:** The APPLY\_TRANSREJECT alert condition detects rejected transactions for this subscription set.

**User Response:** Verify the reason why these transactions were rejected.

---

**ASN5166W** **MONITOR** *monitor\_qualifier*. A manual full refresh is required. The Apply control server is *apply-control-server*. The Apply qualifier is *apply-qualifier*. The name of the subscription set is *set-name*.

**Explanation:** A full refresh is needed for the specified subscription set.

**User Response:** Verify the reason why a full refresh is required.

---

**ASN5167W** **MONITOR** *monitor\_qualifier*. End-to-end latency exceeded the threshold. The Apply control server is *apply-control-server*. The Apply qualifier is *apply-qualifier*. The name of the subscription set is *set-name*. End-to-End latency is *latency seconds*, and the threshold is *threshold seconds*.

**Explanation:** The APPLY\_LATENCY alert condition detects that the end-to-end latency of this subscription set is greater than the threshold value of this alert condition as specified by the PARM\_INT column value in the IBMSNAP\_CONDITIONS table.

**User Response:** Check the Apply control server to determine the reason for this excessive

end-to-end latency value.

---

**ASN5190W** **MONITOR** *monitor\_qualifier*. A stored procedure failed with an SQLCODE of *sqlcode* and with an SQL message of *sql\_message*.

**Explanation:** A user-defined alert condition (implemented as an DB2 stored procedure) returned a nonzero SQLCODE value.

**User Response:** The action depends on the implementation of the stored procedure. If the message indicates an abnormal condition, contact your DBA.

---

**ASN5191W** **MONITOR** *monitor\_qualifier*.  
*message*

**Explanation:** This message appears when a user-defined alert condition issues a warning message.

**User Response:** Read the issued message, and take appropriate action.

---

**ASN5192E** **MONITOR** *monitor\_qualifier*.  
*message*

**Explanation:** This message appears when a user-defined alert condition issues an error message.

**User Response:** Read the issued message, and take appropriate action.

---

**ASN5200E** **ASNSCRT**: The replication process type is a required parameter and must be specified when invoking the *asnsrct* command.

**Explanation:** The *asnsrct* command was invoked without a specified replication process type.

**User Response:** Enter the command again with a replication process type of -C, -A, or -M.

---

**ASN5201E** ASNSCRT: The database instance is a required parameter and must be specified when invoking the asnsCRT command.

**Explanation:** The asnsCRT command was invoked without a specified database instance.

**User Response:** Enter the command again with a database instance name.

---

**ASN5202E** ASNSCRT: A replication process path is a required parameter and must be specified when invoking the asnsCRT command.

**Explanation:** The asnsCRT command was invoked without a specified replication process path.

**User Response:** Enter the command again with a path to the asncap, asnaply, or asnmon command.

---

**ASN5203E** ASNSCRT: The Capture server is a required parameter and must be specified when invoking this asnsCRT command.

**Explanation:** The asnsCRT command was invoked without a specified Capture control server.

**User Response:** Enter the command again with a Capture control server name.

---

**ASN5204E** ASNSCRT: The Apply control server is a required parameter and must be specified when invoking this asnsCRT command.

**Explanation:** The asnsCRT command was invoked without a specified Apply control server.

**User Response:** Enter the command again with an Apply control server name.

---

**ASN5205E** ASNSCRT: The Apply qualifier is a required parameter and must be specified when invoking this asnsCRT command.

**Explanation:** The asnsCRT command was invoked without a specified Apply qualifier.

**User Response:** Enter the command again with an Apply qualifier.

---

**ASN5206E** ASNSCRT: The monitor control server is a required parameter and must be specified when invoking this asnsCRT command.

**Explanation:** The asnsCRT command was invoked without a specified monitor control server.

**User Response:** Enter the command again with a monitor control server name.

---

**ASN5207E** ASNSCRT: The monitor qualifier is a required parameter and must be specified when invoking this asnsCRT command.

**Explanation:** The asnsCRT command was invoked without a specified monitor qualifier.

**User Response:** Enter the command again with a monitor qualifier.

---

**ASN5208I** ASNSCRT: The replication service *service\_name* was created successfully.

**Explanation:** The asnsCRT command successfully created the specified service.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5209I** ASNSCRT: The replication service *service\_name* started successfully.

**Explanation:** The asnsCRT command successfully started the specified service.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5210E** ASNSCRT: The replication service *service\_name* was not created, because the display name already exists (either as a service name or as another display name) in the service control manager database.

**Explanation:** The asnsCRT command cannot create the specified service, because the display name already exists as another service name or display name in the service control manager database.

**User Response:** Go to service control manager database, and remove the service with the duplicate service or display name. Then re-enter the command.

---

**ASN5211E** ASNSCRT: The replication service *service\_name* was not created, because the specified service name is not valid.

**Explanation:** The asnsCRT command cannot create the specified service, because the system API returned an error code indicating that the service name is incorrect. The specified service might contain special characters in the instance name, database name, or schema name. Special characters are not allowed in the service name.

**User Response:** Change the instance name, database name, or schema name if possible. Then re-enter the command.

---

**ASN5212E** ASNSCRT: The replication service *service\_name* was not created, because the specified service name already exists.

**Explanation:** The asnsCRT command cannot create the specified service, because another service with the same service name already exists in the service control manager.

**User Response:** Remove the existing service with the same service name. Then re-enter the command.

---

**ASN5213E** ASNSCRT: The replication service *service\_name* was not started, because the service binary file could not be found.

**Explanation:** The asnsCRT command cannot start the specified service, because the corresponding asncap, asnappl, or asnmon command cannot be invoked using the system path specified by the PATH environment variable. If the fully qualified path is provided, the asnsCRT command cannot find the asncap, asnappl or asnmon command in that path.

**User Response:** Make sure the specified path is correct. Then re-enter the command.

---

**ASN5214E** ASNSCRT: The replication service *service\_name* did not start, because an instance of the service is already running.

**Explanation:** The asnsCRT command cannot start the specified service, because the service is already running.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5215E** ASNSCRT: The replication service *service\_name* did not start, because the service depends on a DB2 instance service that does not exist or has been marked for deletion.

**Explanation:** The asnsCRT command cannot start the specified service, because the corresponding DB2 instance service does not exist or has been deleted.

**User Response:** Verify that the corresponding DB2 instance service exists in the service control manager. Then re-enter the command.

---

**ASN5216E** ASNSCRT: The replication service *service\_name* did not start, because this service depends on another service that failed to start.

**Explanation:** The asnsCRT command cannot start the specified service, because the corresponding

DB2 instance service failed to start.

**User Response:** Verify that the corresponding DB2 instance service started in the service control manager. Then re-enter the command.

---

**ASN5217E** **ASNSCRT: The replication service *service\_name* did not start, because the service is disabled.**

**Explanation:** The asnsCRT command cannot start the specified service, because the service has been disabled in the service control manager.

**User Response:** Verify that the service startup type is set to either automatic or manual in the service control manager. Then re-enter the command.

---

**ASN5218E** **ASNSCRT: The replication service *service\_name* did not start, because the service cannot log on. This error occurs if the service starts from an account that does not have the proper "Log on as a service" access right.**

**Explanation:** The asnsCRT command cannot start the specified service, because the corresponding DB2 instance service cannot log on.

**User Response:** Go to service control manager, and locate the specified service. Verify that the provided account name and passwords are correct. Then re-enter the command.

---

**ASN5219E** **ASNSCRT: The replication service *service\_name* did not start, because the service is marked for deletion.**

**Explanation:** The asnsCRT command cannot start the specified service, because the service has been deleted.

**User Response:** Close the service control manager window. Then re-enter the command.

---

**ASN5220E** **ASNSDROP: The service name is a required parameter and must be specified when invoking the asnsdrop command.**

**Explanation:** The asnsdrop command was invoked without a specified service name.

**User Response:** Re-enter the command with a service name.

---

**ASN5221I** **ASNSDROP: The service *service\_name* has been successfully removed.**

**Explanation:** The asnsdrop command was invoked with a specified service name.

**User Response:** This message is for your information only, and no action is required.

---

**ASN5222E** **ASNSDROP: The replication service *service\_name* cannot be removed, because the requested access is denied.**

**Explanation:** The asnsdrop command cannot remove the specified service name, because the user does not have the appropriate permission to remove it.

**User Response:** Verify that the current user has permission to log on to the corresponding DB2 instance. Then re-enter the command.

---

**ASN5223E** **ASNSDROP: The replication service *service\_name* cannot be removed, because the specified service name is not valid.**

**Explanation:** The asnsdrop command cannot remove the specified service name, because the service name contains illegal special characters.

**User Response:** Go to service control manager, and locate the specified service. Verify that the service name is valid, and re-enter the command.

---

**ASN5224E**    **ASNSDROP: The replication service *service\_name* cannot be removed, because the specified service does not exist.**

**Explanation:** The asnsdrop command cannot remove the specified service name, because the service name does not exist in the service control manager.

**User Response:** Go to service control manager, and locate the specified service. Verify that the service name is correct, and re-enter the command.

---

**ASN5225E**    **ASNSDROP: The replication service *service\_name* cannot be stopped, because other running services are dependent on it. The *service\_name* is not removed.**

**Explanation:** The asnsdrop command cannot remove the specified service, because other services that are dependent on this specified service are currently running.

**User Response:** Go to service control manager, and stop all services that are dependent on this specified service. Then re-enter the command.

---

**ASN5226E**    **ASNSDROP: The replication service *service\_name* cannot be removed, because the system is shutting down.**

**Explanation:** The asnsdrop command cannot remove the specified service, because the operating system is shutting down.

**User Response:** Enter the command again after system restarts.

---

**ASN5227I**    **ASNSDROP: The replication service *service\_name* cannot be removed, because it has already been marked for deletion.**

**Explanation:** The asnsdrop command cannot remove the specified service, because the specified service has already been deleted.

**User Response:** This message is for your

information only, and no action is required.

---

**ASN5228E**    ***pgmname* : The command cannot *command\_action* the replication service *service\_name*, because the system call *API\_func\_name* returned an unexpected error code *error\_code*.**

**Explanation:** The asnsrct and asndrop commands make system calls in order to work with the services. The specified system call returned an unexpected error code that prevents the given command from completing the requested action.

**User Response:** Enter the command again. This error code might indicate only a temporary system condition. For further information about the error code, check your operating system documentation.

---

**ASN5229E**    **ASNSCRT: The account is a required parameter and must be specified when invoking the asnsrct command.**

**Explanation:** The asnsrct command was invoked without a specified account name for the corresponding DB2 instance.

**User Response:** Re-enter the command with an account name for the corresponding DB2 instance.

---

**ASN5230E**    **ASNSCRT: The password is a required parameter and must be specified when invoking the asnsrct command.**

**Explanation:** The asnsrct command was invoked without a specified password for the corresponding DB2 instance.

**User Response:** Re-enter the command with the password for the corresponding DB2 instance.

---

**ASN5231E**    **ASNSCRT: The replication service**  
*service\_name* **did not start, because**  
**the account name specified on the**  
**account parameter does not exist.**

**Explanation:** The asnsCRT command was invoked with an unknown account name for the corresponding DB2 instance.

**User Response:** Go to service control manager, and locate the specified service. Verify that the provided account name and passwords are correct. Then re-enter the command.

---

**ASN5232E**    **ASNSCRT: The required**  
**parameter *path* was not specified.**

**Explanation:** When started as a service, a replication command must contain a path specified by the path keyword that is specific to the command (capture\_path for asncap, apply\_path for asnappl, and monitor\_path for asnmon). If the path keyword is specified, the service is registered if no errors occur.

If the path keyword is not specified, the asnsCRT command retrieves the DB2 global registry profile variable DB2PATH. If this variable contains a non-null value, the asnsCRT command adds the appropriate path keyword to the command using the value of DB2PATH. If this variable is not set, the asnsCRT command cannot register the service.

**User Response:** Enter the command again after you specify the appropriate path keyword or after you define the DB2 global registry profile variable DB2PATH.

---

## Appendix A. UNICODE and ASCII encoding schemes on z/OS

DB2 DataPropagator for OS/390 and z/OS Version 7 or later support both UNICODE and ASCII encoding schemes. To exploit the UNICODE encoding scheme, you must have at least DB2 for OS/390 and z/OS Version 7 and you must manually create or convert your DB2 DataPropagator source, target, and control tables as described in the following sections. However, your existing replication environment will work with DB2 DataPropagator for OS/390 and z/OS Version 7 or later even if you do not modify any encoding schemes. If your system is a UNICODE system, you must add ENCODING(EBCDIC) on the BIND PLAN and PACKAGE commands for the Capture, Apply, and Replication Alert Monitor programs.

---

### Choosing an Encoding Scheme

If your source, CD, and target tables use the same encoding scheme, you can minimize the need for data conversions in your replication environment. When you choose encoding schemes for the tables, follow the single CCSID rule:

The table space data is encoded using ASCII or EBCDIC or UNICODE CCSIDs. All tables within a table space must use the same encoding scheme. The encoding scheme of all the tables referenced by an SQL statement must be the same. Also, all tables that you use in views and joins must use the same encoding scheme.

If you do not follow the single CCSID rule, DB2 will detect the violation and return SQLCODE -873 during bind or execution.

Which tables should be ASCII or UNICODE depends on your client/server configuration. Specifically, follow these rules when you choose encoding schemes for the tables:

- Source or target tables on DB2 for OS/390 can be EBCDIC, ASCII, or UNICODE. They can be copied from or to tables that have the same or different encoding scheme in any supported DBMS (DB2 family, or non-DB2 with DataJoiner).
- On a DB2 for OS/390 source server, CD and UOW tables on the same server do not have to use the same encoding scheme if, when the subscription set is created, the target type is USERCOPY and JOIN\_UOW\_CD=Y. Otherwise, the CD and UOW tables must use the same encoding scheme.

- The signal (IBMSNAP\_SIGNAL) table should be encoded EBCDIC so that the Capture program does not have to translate signals to EBCDIC when it selects them from the signal table.
- All the control tables (ASN.IBMSNAP\_SUBS\_XXXX) on the same control server must use the same encoding scheme.
- Other control tables can use any encoding scheme.

---

## Setting Encoding Schemes

To specify the proper encoding scheme for tables, modify the SQL that is used to generate the tables.

1. Create new source and target tables with the proper encoding scheme, or change the encoding schemes of the existing target and source tables. It is recommended that you stop the Capture and Apply programs before you change the encoding scheme of existing tables, and afterwards that you could start the Capture program and restart the Apply program. To change the encoding scheme of existing tables (which must have the same encoding scheme within a table space):
  - a. Use the Reorg Tablespace utility to unload the existing table space.
  - b. Drop the existing table space.
  - c. Re-create the table space specifying the new encoding scheme.
  - d. Use the Load utility to load the old data into the new table space.

See the *DB2 Universal Database for OS/390 and z/OS Utility Guide and Reference*, SC26-9945 for more information on the Load and Reorg utilities.

2. Use the Replication Center to create new control tables with the proper encoding scheme.
3. Use the Reorg and Load utilities to modify the encoding scheme for existing control tables and CD tables.
4. When you create new replication sources or subscription sets using the Replication Center, specify the proper encoding scheme.

The *SQL Reference*, SC26-9014 contains more information about CCSID.



---

## Appendix B. How the Capture program processes journal entry types (iSeries)

The following table describes how the Capture program processes different journal entry types.

*Table 100. Capture program processing by journal entry*

<b>Journal code<sup>3</sup></b>	<b>Entry type</b>	<b>Description</b>	<b>Processing</b>
C	CM	Set of record changes committed	Insert a record in the UOW stable.
C	RB	Rollback	No UOW row inserted.
F	AY	Journalized changes applied to physical file member	Issue an ASN2004 message and full refresh of file.
F	CE	Change end of data for physical file	Issue an ASN2004 message and full refresh of file.
F	CR	Physical file member cleared	Issue an ASN2004 message and full refresh of file.
F	EJ	Journaling for physical file member ended	Issue an ASN2004 message and full refresh of file.
F	IZ	Physical file member initialized	Issue an ASN2004 message and full refresh of file.
F	MD	Member removed from physical file (DLTLIB, DLTF, or RMVM)	Issue an ASN2004 message and attempt a full refresh.
F	MF	Storage for physical file member freed	Issue an ASN2004 message and full refresh of file.
F	MM	Physical file containing member moved (Rename Object (RNMOBJ) of library, Move Object (MOV OBJ) of file)	Issue an ASN200A message and attempt a full refresh.
F	MN	Physical file containing member renamed (RNMOBJ of file, Rename Member (RNMM))	Issue an ASN200A message and attempt a full refresh.
F	MR	Physical file member restored	Issue an ASN2004 message and full refresh of file.

Table 100. Capture program processing by journal entry (continued)

Journal code <sup>3</sup>	Entry type	Description	Processing
F	RC	Journalized changes removed from physical file member	Issue an ASN2004 message and full refresh of file.
F	RG	Physical file member reorganized	If the RRN of the source table is being used as the replication key, issue an ASN2004 message and full refresh of file.
J	NR	Identifier for next journal receivers	Reset the Capture program.
J	PR	Identifier for previous journal receivers	Increment the unique sequence number counter.
R	DL	Record deleted from physical file member	Insert a DLT record in the CD table.
R	DR	Record deleted for rollback	Insert a DLT record in the CD table.
R	PT	Record added to physical file member	Insert an ADD record in the CD table.
R	PX	Record added directly to physical file member	Insert an ADD record in the CD table.
R	UB	Before-image of record updated in physical file member	See note 1.
R	UP	After-image of record updated in physical file member	See note 1.
R	BR	Before-image of record updated for rollback	See note 2.
R	UR	After-image of record updated for rollback	See note 2.

Table 100. Capture program processing by journal entry (continued)

Journal code <sup>3</sup>	Entry type	Description	Processing
<b>Notes:</b>			
1. The R-UP image and the R-UB image form a single UPD record in the CD table if the PARTITION_KEYS_CHG column in the register table is N. Otherwise, the R-UB image inserts a DLT record in the CD table and the R-UP image inserts an ADD record in the CD table.			
2. The R-UR image and the R-BR image form a single UPD record in the CD table if the PARTITION_KEYS_CHG column in the register table is N. Otherwise, the R-BR image inserts a DLT record in the CD table and the R-UR image inserts an ADD record in the CD table.			
3. The following values are used for the journal codes:			
C Commitment control operation			
F Database file operation			
J Journal or journal receiver operation			
R Operation on specific record			

All other journal entry types are ignored by the Capture program.



---

## Appendix C. Starting the replication programs from within an application (UNIX, Windows)

You can start any of the replication programs (Capture program, Apply program, Replication Alert Monitor) for one replication cycle from within your application by calling routines. To use these routines you must specify the AUTOSTOP option for the Capture program and the COPYONCE option for the Apply program because the API support only synchronous execution.

Samples of the API and their respective makefiles are in the following directories:

**For NT**

sqllib\samples\repl

**For UNIX**

sqllib/samples/repl

Those directories contain the following files for starting the Capture program:

**capture\_api.c**

The sample code for starting the Capture program on Windows or UNIX.

**capture\_api\_nt.mak**

The makefile for sample code on Windows.

**capture\_api\_unix.mak**

The makefile for sample code on UNIX.

Those directories contain the following files for starting the Apply program:

**apply\_api.c**

The sample code for starting the Apply program on Windows or UNIX.

**apply\_api\_nt.mak**

The makefile for sample code on Windows.

**apply\_api\_unix.mak**

The makefile for sample code on UNIX.

Those directories contain the following files for starting the Replication Alert Monitor:

**monitor\_api.c**

The sample code for starting the Replication Alert Monitor on Windows or UNIX.

**monitor\_api\_nt.mak**

The makefile for sample code on Windows.

**monitor\_api\_unix.mak**

The makefile for sample code on UNIX.

---

## Glossary

### A

**after-image.** The updated content of a source-table column that is recorded in a change data (CD) table, or in a database log or journal. Contrast with *before-image*.

**aggregate table.** A read-only replication target table that contains aggregations of data from the source table. This data is based on SQL column functions such as MIN, MAX, SUM, or AVG.

**alert condition.** A user-specified condition for monitoring replication, that, when met, causes the Replication Alert Monitor to send notification to an individual or group that an error occurred or an operational threshold was reached.

**Apply control server.** A database that contains the Apply control tables, which store information about subscription sets and subscription-set members. Contrast with *Apply server*.

**Apply cycle.** The interval of time during which data is replicated from a source table to a target table.

**Apply latency.** An approximate measurement of the time that replication requires to complete one cycle. See also *Capture latency*.

**Apply program.** A program that is used to refresh or update a replication target table, depending on the applicable source-to-target rules. Contrast with *Capture program* and *Capture trigger*.

**Apply qualifier.** A case-sensitive character string that identifies replication subscription sets that are unique to an instance of the Apply program.

**Apply server.** A system where the Apply program is running. Contrast with *Apply control server*.

**archive log.** (1) The set of log files that is closed and is no longer needed for normal processing. These files are retained for use in roll-forward recovery. (2) The portion of the DB2 Universal Database for z/OS and OS/390 log that contains log records that are copied from the active log. The archive log holds records that are older and no longer fit on the active log.

**asynchronous replication.** The process of copying data from a source table to a target table outside the scope of the original transaction that updated the source table. Contrast with *synchronous replication*.

**audit trail.** Data, in the form of a logical path that links a sequence of events, used for tracing the transactions that affected the contents of a record.

**authorization token.** (1) A token associated with a transaction. (2) For DB2 Universal Database for z/OS, the correlation ID. (3) For DB2 Universal Database for iSeries, the job name of the job that caused a transaction.

### B

**base aggregate table.** A type of replication target table that contains data that is aggregated from a replication source table. It includes a timestamp to mark the time when the Apply program performed the aggregation. Contrast with *change aggregate table*.

**before-image.** The content of a replication source-table column prior to an update by a transaction, as recorded in a change data (CD) table, or in a database log or journal. Contrast with *after-image*.

**binary large object (BLOB).** A sequence of bytes with a size from 0 bytes to 2 gigabytes less 1 byte. This string does not have an associated code page and character set. BLOBs can contain

image, audio, and video data. See also *character large object* and *double-byte character large object*.

**BLOB.** See *binary large object*.

**block fetch.** A function of DB2 that retrieves (or fetches) a large set of rows together. Using block fetch can significantly reduce the number of messages that are sent across the network. Block fetch applies only to cursors that do not update data.

**blocking.** An option that is specified when binding an application. It allows caching of multiple rows of information by the communications subsystem so that each FETCH statement does not require the transmission of one row for each request across the network. See also *block fetch*.

## C

**Capture control server.** (1) A database that contains the Capture control tables, which store information about registered replication source tables. (2) A system where the Capture program is running.

**Capture latency.** An approximate measurement of how recently the Capture program committed data to a CD table. See also *Apply latency*.

**Capture program.** A program that reads database log or journal records to capture changes made to DB2 source tables. Contrast with *Apply program* and *Capture trigger*.

**Capture schema.** The schema for the Capture control tables used by a particular instance of the Capture program.

**Capture trigger.** A mechanism that captures delete, insert, and update operations performed on non-DB2 relational source tables. Contrast with *Capture program* and *Apply program*.

**cascade rejection.** The process of rejecting a replication transaction because it is associated with a transaction that had a conflict detected and was itself rejected.

**CCD table.** See *consistent-change data (CCD) table*.

**CD table.** See *change data table*.

**change aggregate table.** A type of replication target table that contains data aggregations based on the contents of a CD table. It includes two timestamps to mark the time interval for when the changes were written to the CD table. Contrast with *base aggregate table*.

**change-capture replication.** The process of capturing changes made to a replication source table and applying them to a replication target table. Contrast with *full refresh*.

**change data (CD) table.** A replication table at the Capture control server that contains changed data for a replication source table.

**character large object (CLOB).** A sequence of characters (single-byte, multibyte, or both) with a size ranging from 0 bytes to 2 gigabytes less 1 byte. In general, character large object values are used whenever a character string might exceed the limits of the VARCHAR type. Also called character large object string. See also *binary large object* and *double-byte character large object*.

**client.** Any program (or workstation that a program is running on) that communicates with and accesses a database server.

**CLOB.** Character large object.

**cold start.** (1) For DB2 replication, the process of starting the Capture program without using restart information from prior operation of the Capture program. Contrast with *warm start*. (2) The process of starting a system or program by using an initial program load procedure. (3) A process by which DB2 Universal Database for z/OS and OS/390 restarts without processing any log records.

**complete CCD table.** A CCD table that initially contains all the rows from the replication source table or view and any predicates from the source table or view. Contrast with *noncomplete CCD table*. See also *consistent-change data (CCD) table*.



**condensed.** A table attribute indicating that the table contains current data rather than a history of changes to the data. A condensed table includes no more than one row for each primary key value in the table. As a result, a condensed table can be used to supply current information for a refresh.

**condensed CCD table.** A CCD table that contains only the most current value for a row and has only one row for each key value. Contrast with *noncondensed CCD table*. See also *consistent-change data (CCD) table*.

**conflict detection.** In update-anywhere replication configurations, conflict detection refers to either of the following processes:

- The process of detecting constraint errors.
- The process of detecting whether the same row was updated by users or application programs in both the source and target tables during the same replication cycle. When a conflict is detected, the transaction that caused the conflict is rejected.

**consistent-change data (CCD) table.** A type of replication target table that is used for storing history, for auditing, or for staging data. A CCD table can also be a replication source. See also *complete CCD table*, *condensed CCD table*, *external CCD table*, *internal CCD table*, *noncomplete CCD table*, and *noncondensed CCD table*.

**Control Center.** The DB2 graphical interface that shows database objects (such as databases and tables) and their relationship to each other. From the Control Center, you can perform the tasks provided by the DBA Utility, Visual Explain, and Performance Monitor tools.

**control server.** A database server that contains replication control tables for the Capture, Apply, or Monitor program. See also *Apply control server*, *Capture control server*, and *Monitor control server*.

**control table.** See *replication control table*.

## D

**database log.** A set of primary and secondary log files consisting of log records that record all changes to a database. The database log is used to roll back changes for transactions (units of work) that are not committed and to recover a database to a consistent state.

**database management system (DBMS).**  
Synonym for *database manager*.

**database manager.** A computer program that manages data by providing the services of centralized control, data independence, and complex physical structures for efficient access, integrity, recovery, data currency control, privacy, and security.

**database server.** The target of a request from a local application or an intermediate database server. In the DB2 environment, the database server function is provided by the distributed data facility to access DB2 data from local applications or a remote database server that is acting as an intermediate database server.

**data blocking.** The process of replicating a specific number of minutes' worth of change data during an Apply cycle.

**data consolidation.** A replication configuration that contains one read-only target database. The target table contains rows of data from one or more source tables.

**data distribution.** A replication configuration that contains a single source table, from which changes are replicated to one or more read-only target tables. Before replication to the target tables can occur, the tables must contain a complete set of data from the source table.

**DBCLOB.** Double-byte character large object.

**DBMS.** Database management system.

**delimited identifier.** A sequence of characters enclosed within quotation marks ("). The sequence must consist of a letter followed by

zero or more characters, each of which is a letter, a digit, or the underscore character. See also *ordinary identifier*.

**differential refresh.** See *change-capture replication*.

**distinct type.** A user-defined data type that is internally represented as an existing type (its source type), but is considered to be a separate and incompatible type for semantic purposes. See also *user-defined type (UDT)*.

**double-byte character large object (DBCLOB).** A sequence of double-byte characters, with a size ranging from 0 bytes to 2 gigabytes. This data type can be used to store large double-byte text objects. Also called *double-byte character large object string*. Such a string always has an associated code page.

## E

**end-to-end latency.** An approximate measurement of the time that replication requires to capture changes and apply those changes to a target database. See also *Apply latency* and *Capture latency*.

**event timing.** The most precise method of controlling when to start a replication subscription cycle. To use event timing, you must specify an event name and the time that you want the event to be processed. Contrast with *interval timing*.

**external CCD table.** A CCD table that can be subscribed to directly because it is a registered replication source. It has its own row in the register table, where it is identified by the SOURCE\_OWNER and SOURCE\_TABLE columns. See also *consistent-change data table*. Contrast with *internal CCD table*.

## F

**federated database system.** A special type of distributed database management system (DBMS). A federated system allows you to query and manipulate data located on other servers.

The data can be in database managers such as Oracle, Sybase, and Microsoft SQL Server, or it can be in lists or stores such as a spreadsheet, Web site, or data mart. SQL statements can refer to multiple database managers or individual databases in a single statement. For example, you can join data located in a DB2 Universal Database table, an Oracle table, and a Sybase view.

**full refresh.** The process in which all of the data that matches registration and subscription-set predicates for a replication source table is copied to the target table. A full refresh replaces all existing data in the target table. In a data distribution configuration, a full refresh must be complete before any other data is replicated. Contrast with *change-capture replication*.

## G

**gap.** A situation in which the Capture program is not able to read a range of log or journal records, so there is potential loss of change data.

**global record.** The row in the register table that defines global replication characteristics for a particular instance of the Capture program.

## H

**heterogeneous replication.** Replication between DB2 and non-DB2 relational databases. See also *federated database system*.

**hot-spot update.** A series of repeated updates made to the same rows in a short period of time.

## I

**internal CCD table.** A CCD table that cannot be subscribed to directly because it is not a registered replication source. It does not have its own row in the register table; it is identified by the CCD\_OWNER and CCD\_TABLE columns for the row of the associated registered replication source. Contrast with *external CCD table*. See also *consistent-change data (CCD) table*.

**interval timing.** The simplest method of controlling when to start a replication subscription cycle. To use interval timing, you must specify a date and a time for a subscription cycle to start, and set a time interval that describes how frequently you want the subscription cycle to run. Contrast with *event timing*.

## J

**join.** An SQL relational operation that allows retrieval of data from two or more tables based on matching column values.

**journal.** For iSeries systems, a system object that identifies the objects being journaled, the current journal receiver, and all the journal receivers on the system for the journal. See also *journal receiver*.

**journal code.** For iSeries systems, a 1-character code in a journal entry that identifies the category of the journal entry. For example, F identifies an operation on a file; R identifies an operation on a record, and so forth. See also *journal entry*.

**journal entry.** For iSeries systems, a record in a journal receiver that contains information about a journaled change or other activity that is journaled. See also *journal code* and *journal entry type*.

**journal entry type.** For iSeries systems, a 2-character field in a journal entry that identifies the type of operation of a system-generated journal entry or the type of journal entry of a user-generated journal entry; for example, PT is the entry type for a write operation. See also *journal code*.

**journal identifier (JID).** For iSeries systems, a unique identifier that is assigned to a particular object when journaling is started for that object. Journal entries are associated with a particular object by this JID value.

**journaling.** For iSeries systems, the process of recording, in a journal, the changes made to objects, such as physical file members or access

paths, or the depositing of journal entries by system or user functions.

**journal receiver.** For iSeries systems, a system object that contains journal entries added when events occur that are journaled, such as changes to a database file, changes to other journaled objects, or security-relevant events. See also *journal*.

## K

**key.** A column or an ordered collection of columns that is identified in the description of a table, index, or referential constraint. The same column can be part of more than one key.

## L

**large object (LOB).** A sequence of bytes with a size ranging from 0 bytes to 2 gigabytes less 1 byte. It can be any of three types: binary large object (binary), character large object (single-byte character or mixed), or double-byte character large object (double-byte character).

**latency.** The time required for updates made to a replication source to appear in a replication target.

**LOB.** See *large object*.

**local database.** A database that is physically located on the workstation in use. Contrast with *remote database*.

**lock.** (1) A means of serializing events or access to data. (2) A means of preventing uncommitted changes made by one application process from being perceived by another application process and for preventing one application process from updating data that is being accessed by another process.

**locking.** The mechanism used by the database manager to ensure the integrity of data. Locking prevents concurrent users from accessing inconsistent data.

**log.** (1) A file used to record changes made in a system. (2) A collection of records that describe

the events that occur during DB2 Universal Database for OS/390 execution and that indicate their sequence. The information thus recorded is used for recovery in the event of a failure during DB2 Universal Database for OS/390 execution.  
(3) See *database log*.

## M

**master table.** In update-anywhere replication, the original source table for data in the replica table. If replication conflict detection is enabled, changes made to the master table are retained, whereas changes made to the replica table are rejected. See also *update-anywhere replication*, *replica table*, and *conflict detection*.

**member.** See *subscription-set member*.

**Monitor control server.** A database that contains the replication Monitor control tables, which store information about alert conditions that replication will monitor.

**Monitor qualifier.** A case-sensitive character string that identifies an instance of a replication Monitor process.

**multi-tier replication.** A replication configuration in which changes are replicated from a replication source in one database to a replication target in another database, and changes from this replication target are replicated again to a replication target in another database.

## N

**nickname.** (1) An identifier that a federated server uses to reference a data source object, such as a table or a view. (2) A name that is defined in a DB2 DataJoiner database to represent a physical database object (such as a table or stored procedure) in a non-DB2 database.

**noncomplete CCD table.** A CCD table that is initially empty and has rows appended to it as changes are made to the replication source. Contrast with *complete CCD table*. See also *consistent-change data (CCD) table*.

**noncondensed CCD table.** A CCD table that can contain more than one row for each key value. These duplicate rows represent the history of changes for the values in rows of a table. Contrast with *condensed CCD table*. See also *consistent-change data (CCD) table*.

**non-DB2 relational database server.** An Informix database server or a relational database server from a vendor other than IBM.

**nullable.** The condition in which a value for a column, function parameter, or result can have an absence of a value.

**null value.** A parameter position for which no value is specified.

## O

**object.** (1) Anything that can be created or manipulated with SQL—for example, tables, views, indexes, or packages. (2) In object-oriented design or programming, an abstraction that consists of data and operations associated with that data. (3) For NetWare, an entity that is defined on the network and thus given access to the file server. (4) In the Information Catalog Center, an item that represents a unit or distinct grouping of information. Each Information Catalog Center object identifies and describes information but does not contain the actual information. For example, an object can provide the name of a report, list its creation date, and describe its purpose.

**occasionally connected.** A replication configuration that contains target servers which are not always connected to the network. This configuration allows users to connect to a primary data source for a short time to synchronize their local database with the data at the source.

**ODBC.** See *Open Database Connectivity (ODBC)*.

**ODBC driver.** A driver that implements ODBC function calls and interacts with a data source.

**Open Database Connectivity (ODBC).** An API that allows access to database management

systems using callable SQL, which does not require the use of an SQL preprocessor. The ODBC architecture allows users to add modules, called *database drivers*, that link the application to their choice of database management systems at run time. Application programs do not need to be linked directly to the modules of all the supported database management systems.

**ordinary identifier.** (1) In SQL, a letter followed by zero or more characters, each of which is a letter (a-z and A-Z), a symbol, a number, or the underscore character, used to form a name. (2) In DB2 Universal Database for z/OS and OS/390, an *uppercase* letter followed by zero or more characters, each of which is an *uppercase* letter, a number, a digit, or the underscore character.

## P

**package.** A control structure produced during program preparation that is used to execute SQL statements.

**peer table.** A replication source or target table defined as part of a peer-to-peer replication configuration.

**peer-to-peer replication.** A replication configuration in which all peer tables are both registered sources and read-write targets, and there is no primary source table for full refresh. In this configuration, there is no replication hierarchy among the peer tables. Contrast with *update-anywhere replication*. See also *multi-tier replication*.

**point-in-time table.** A type of replication target table whose content matches all or part of a source table, with an added column that identifies the approximate time when the particular row was inserted or updated at the source system.

**predicate.** An element of a search condition that expresses or implies a comparison operation.

**primary key.** A unique key that is part of the definition of a table. A primary key is the default parent key of a referential constraint definition. It

is a column or combination of columns that uniquely identifies a row in a table.

**promote.** To copy replication definitions for subscription sets or registered sources from one database to another database, without having to re-register the sources or re-create the subscription sets.

**pruning.** The task of removing obsolete data from replication control tables, CD tables, CCD tables, or Capture or Apply log files.

**pull configuration.** A replication configuration in which the Apply program runs at the target server; the Apply program pulls updates from the source server to apply them to the target. Contrast with *push configuration*.

**push configuration.** A replication configuration in which the Apply program runs at the source server or a replication server other than the target server; the Apply program pushes updates from the source server to apply them to the target. Contrast with *pull configuration*.

## R

**RDBMS.** See *relational database management system (RDBMS)*.

**real-time replication.** See *synchronous replication*.

**recapture.** In update-anywhere replication, to capture changes at a replica table and forward these changes to the master table or to other replica tables.

**referential constraints.** The referential integrity rule that the non-null values of the foreign key are valid only if they also appear as values of a parent key.

**referential integrity.** The state of a database in which all values of all foreign keys are valid. Maintaining referential integrity requires the enforcement of *referential constraints* on all operations that change the data in a table upon which the referential constraints are defined.

**registration.** (1) The process of registering a DB2 table, view, or nickname as a replication source. Contrast with *subscription*. (2) See *replication source*.

**rejected transaction.** A transaction containing one or more updates from replica tables that are in conflict with the master table.

**relational database management system (RDBMS)** . A collection of hardware and software that organizes and provides access to a relational database.

**remote database.** A database that is physically located on a workstation other than the one in use. Contrast with *local database*.

**replica table.** In update-anywhere replication, a type of target table that can be updated locally and also receives updates from the master table through a subscription-set definition. If replication conflict detection is enabled, changes made to the replica table are rejected, whereas changes made to the master table are retained. See also *update-anywhere replication*, *master table*, and *conflict detection*.

**replication.** The process of maintaining a defined set of data in more than one location. It involves copying designated changes for one location (a source) to another (a target), and synchronizing the data in both locations.

**replication administrator.** The user responsible for registering replication sources and creating subscription sets. This user can also run the Capture and Apply programs.

**Replication Alert Monitor.** A set of programs that can monitor the activity of the Capture and Apply programs, and depending on user-defined alert conditions, can send notifications to specific users.

**Replication Analyzer.** A program that can analyze a replication environment for setup problems, configuration errors, and performance issues.

**Replication Center.** A graphical user interface for DB2 Replication that shows Capture and

Apply control servers, registered sources, subscription sets, and Monitor control servers. From the Replication Center, a replication administrator can also perform operational tasks for the Capture and Apply programs.

**replication control table.** A table in which replication definitions or control information is stored.

**replication source.** A database table, view, or nickname that is registered as a source for replication. Changes made to this type of table are captured and copied to a target table defined in a subscription-set member. See also *subscription set* and *subscription-set member*.

**retention-limit pruning.** The pruning of CD and UOW tables by the Capture program that are older than a user-specified limit.

**rework.** (1) To convert an insert into a replication target table to an update if the insert fails because the row already exists in the target table. (2) To convert an update to a replication target table to an insert if the update fails because the row does not exist in the target table.

**row-capture rules.** Rules based on changes to registered columns that define when and whether the Capture program writes a row to a CD table, or when and whether the Capture triggers write a row to a CCD table.

## S

**serialization.** (1) The consecutive ordering of items. (2) The process of controlling access to a resource to protect the integrity of the resource.

**server.** See *database server*.

**signal.** A communication mechanism for replication that allows the Capture, Apply, and Monitor programs to communicate with each other asynchronously.

**source server.** A database that contains registered replication sources.

**source table.** A table that contains data that is to be replicated to a target table. Contrast with *target table*.

**spill file.** A temporary file created by the Apply program that is used to hold data for updating target tables.

**staging table.** A CCD table that is used to save data before that data is replicated to the target database. A CCD table used for staging data can function as an intermediate source for updating data to one or more target tables. See also *consistent-change-data table*.

**subscription.** (1) The process of creating subscription sets and subscription-set members. Contrast with *registration*. (2) See *subscription set*.

**subscription cycle.** The process in which the Apply program retrieves changed data for a given subscription set, replicates the changes to the target table, and updates the appropriate replication control tables to reflect its status and current progress.

**subscription set.** A replication definition that controls the replication of changed data during a subscription cycle. A subscription set can contain zero or more subscription-set members.

**subscription-set member.** A replication definition that maps a registered replication source with a replication target. Each member defines the structure of the target table and the rows and columns that will be replicated from the source table.

**subset.** To replicate data from part of a source table, rather than from the entire table, to a target table. You can subset by rows or by columns.

**synchpoint.** A replication control table value for the DB2 log or journal record sequence number of the last change applied during the most recent Apply cycle. This value is also used to coordinate the pruning of CD tables.

**synchronous replication.** Also known as real-time replication, a type of replication that

delivers updates continuously and within the scope of source transactions.

## T

**table-mode processing.** A type of replication subscription-set processing in which the Apply program retrieves all the data from the source CD table, then applies the data (one member at a time) to each target table, and finally commits its work. Contrast with *transaction-mode processing*.

**target server.** A database that contains replication target tables.

**target table.** A table that is the target for replicated changes from a registered replication source. It can be a user copy table, a point-in-time table, a base aggregate table, a change aggregate table, a CCD table, or a replica table.

**temporary table.** A table that holds temporary data. For example, temporary tables are useful for holding or sorting intermediate results from queries that contain a large number of rows. The two kinds of temporary tables, which are created by different SQL statements, are the created temporary table and the declared temporary table.

**timestamp.** A seven-part value that consists of a date and time expressed in years, months, days, hours, minutes, seconds, and microseconds.

**trace.** (1) A DB2 Universal Database for z/OS and OS/390 facility that provides the ability to monitor and collect monitoring, auditing, performance, accounting, statistics, and serviceability (global) data. (2) For DB2 replication, a facility that provides the ability collect monitoring, auditing, and performance data for the Capture program, Apply program, or Replication Alert Monitor.

**transaction.** (1) An exchange between a workstation and a program, two workstations, or two programs that accomplishes a particular action or result. An example of a transaction is the entry of a customer's deposit and the subsequent update of the customer's balance.

Synonym for *unit of work*. (2) One Net.Data invocation. If persistent Net.Data is used, then a transaction can span multiple Net.Data invocations.

**transaction-based replication.** A type of replication processing in which every transaction is replicated to the target table when it is committed in the source table. Contrast with *transaction-consistent replication*.

**transaction-consistent replication.** A type of replication processing in which the net result of all transaction updates is replicated to the target table. Contrast with *transaction-based replication*.

**transaction-mode processing.** A type of replication subscription-set processing in which the Apply program retrieves data from the source CD table, then applies the data to the target table in the same commit sequence used at the source. The Apply program processes transactions for all subscription-set members together, rather than sequentially. Contrast with *table-mode processing*.

**trigger.** (1) An object in a database that is invoked indirectly by the database manager when a particular SQL statement is run. See also *Capture trigger*. (2) A set of SQL statements that is stored in a DB2 database and executed when a certain event occurs in a DB2 table.

## U

**UDT.** See *user-defined type (UDT)*.

**uncommitted read (UR).** An isolation level that allows an application to access uncommitted changes of other transactions. The application does not lock other applications out of the row that it is reading unless the other application attempts to drop or alter the table.

**Unicode.** An international character encoding scheme that is a subset of the ISO 10646 standard. Each character supported is defined using a unique 2-byte code.

**unique index .** An index that ensures that no identical key values are stored in a table.

**unique key.** A key that is constrained so that no two of its values are equal.

**unit of work.** (1) A recoverable sequence of operations within an application process. At any time, an application process is a single unit of work, but the life of an application process can involve many units of work as a result of commit or rollback operations. In a DB2 Universal Database for z/OS and OS/390 multisite update operation, a single unit of work can include several units of recovery. Synonym for *transaction*. (2) In the Information Catalog Center, a recoverable sequence of operations within an application process. At any time, an application process is a single unit of work, but the life of an application process can involve many units of work as a result of commit or rollback operations.

**unit-of-work (UOW) table.** A replication control table stored in the Capture control server that contains commit records read from the database log or journal. The records include a unit-of-recovery ID that can be used to join the unit-of-work table and the CD table to produce transaction-consistent change data.

**update-anywhere replication.** A replication configuration in which all tables are both registered sources and read-write targets. One table is the primary source table for full refresh of all the others. In this configuration, there is an implicit replication hierarchy among the source and target tables. Contrast with *peer-to-peer replication*. See also *multi-tier replication*, *master table*, and *replica table*.

**user copy table.** A replication target table whose content matches all or part of a registered source table and contains only user data columns.

**user-defined type (UDT).** A data type that is not native to the database manager and was created by a user. In DB2 Universal Database, the term *distinct type* is used instead of user-defined type.



## V

**view.** (1) A logical table that consists of data that is generated by a query. A view is based on an underlying set of base tables, and the data in a view is determined by a select type query that is run on the base tables. (2) A way of looking at the information about, or contained in objects. Each view might reveal different information about its objects.

## W

**warm start.** (1) A restart that allows reuse of previously initialized input and output work queues. (2) For DB2 replication, the process of starting the Capture program using existing data from the Capture control tables. Contrast with *cold start*.

**work file.** A temporary file used by the Apply program when processing a subscription set.



---

# Index

## Special Characters

; delimiter 116  
\$TA JES2 command 464  
\*.APP.log file 144  
\*.CAP.log file 126  
\*.err file 147  
\*.sqz file 147, 148  
# delimiter 116

## A

abstract data types 97  
activating subscription sets 68, 261  
ADDDPRREG command 349  
ADDDPRSUB command 359  
ADDDPRSUBM command 376  
ADDEXITPGM command 36  
ADDJOBSCDE command 464  
administration  
    authorization requirements 17  
after-image columns 44  
aggregate tables  
    base aggregate 81, 541  
    change aggregate 82, 542  
alert conditions  
    for Replication Alert Monitor 171  
Alert Monitor  
    See Replication Alert Monitor  
alert\_prune\_limit parameter 175  
ALERTS (Monitor alerts) table 533  
ALWINACT parameter 432  
Analyzer  
    for OS/400  
        creating SQL packages 31  
        invocation parameters 388  
    for UNIX, invocation parameters 306  
    for Windows, invocation parameters 306  
Analyzer report  
    ANZDPR command 387  
    asnanalyze command 305  
ANZDPR command 387  
ANZDPRJRN command 35  
APPENQ (Apply enqueue)  
    table 513  
APPLHEAPSZ configuration  
    parameter 27  
applications  
    starting replication programs  
        from 647  
Apply control server  
    adding to Replication Center 254  
    control tables at 513  
Apply control tables  
    APPENQ (Apply enqueue) 513  
    APPLY\_JOB (Apply job) 513  
    APPLYTRACE (Apply trace) 514  
    APPLYTRAIL (Apply trail) 515  
    list of 513  
    SUBS\_COLS (subscription columns) 519  
    SUBS\_EVENT (subscription events) 521  
    SUBS\_MEMBR (subscription members) 522  
    SUBS\_SET (subscription sets) 526  
    SUBS\_STMTS (subscription statements) 531  
Apply enqueue (APPENQ)  
    table 513  
Apply job (APPLY\_JOB) table 513  
Apply program  
    authorization requirements 21  
    commands 303  
    communicating with  
        Capture program 465, 466  
        Capture triggers 465, 468  
        Replication Alert Monitor 470  
        Replication Center 465  
    connectivity 15  
    data blocking 69  
    for OS/400  
        ALWINACT parameter 432  
        APYQUAL parameter 430  
        checking status 163  
        COPYONCE parameter 433  
        creating SQL packages 31  
        CTLSVR parameter 430  
        DELAY parameter 433  
        FULLREFPGM parameter 431  
        INACTMSG parameter 432

Apply program (*continued*)  
    for OS/400 (*continued*)  
        JOB parameter 429  
        OPTSNGSET parameter 434  
        RTYWAIT parameter 433  
        scheduling 464  
        setting up 30, 32  
        starting 149, 427  
        stopping 151, 397  
        SUBNFYPGM parameter 432  
        TRACE parameter 431  
        TRLREUSE parameter 434  
        USER parameter 429  
    for UNIX  
        apply\_path parameter 141, 310  
        apply\_qual parameter 141, 304, 310  
        binding 28  
        checking status 161  
        configuring 28  
        control\_server  
            parameter 142, 304, 310  
        copyonce parameter 142, 312  
        delay parameter 143, 313  
        errwait parameter 143, 313  
        inamsg parameter 144, 311  
        loadxit parameter 144, 311  
        logreuse parameter 144, 311  
        logstdout parameter 145, 311  
        notify parameter 145, 312  
        operating 304  
        opt4one parameter 145, 313  
        password file 23  
        pwdfile parameter 146, 310  
        setting up 26  
        sleep parameter 146, 312  
        spillfile parameter 147, 315  
        sqlerrorcontinue  
            parameter 147, 314  
        starting 139, 308, 647  
        status 304  
        stopping 151, 304  
        term parameter 148, 314  
        trlreuse parameter 148, 313  
    for Windows  
        apply\_path parameter 141, 310

Apply program (*continued*)  
 for Windows (*continued*)  
   apply\_qual parameter 141, 304, 310  
   binding 28  
   checking status 161  
   configuring 28  
   control\_server  
     parameter 142, 304, 310  
   copyonce parameter 142, 312  
   delay parameter 143, 313  
   errwait parameter 143, 313  
   inamsg parameter 144, 311  
   loadxit parameter 144, 311  
   logreuse parameter 144, 311  
   logstdout parameter 145, 311  
   notify parameter 145, 312  
   operating 139, 304  
   opt4one parameter 145, 313  
   password file 23  
   pwdfile parameter 146, 310  
   setting up 26  
   sleep parameter 146, 312  
   spillfile parameter 147, 315  
   sqlerrorcontinue  
     parameter 147, 314  
   starting 139, 308, 647  
   status 304  
   stopping 151, 304  
   term parameter 148, 314  
   trlreuse parameter 148, 313  
 for z/OS  
   apply\_path parameter 141, 310  
   apply\_qual parameter 141, 304, 310  
   checking status 161  
   control\_server  
     parameter 142, 304, 310  
   copyonce parameter 142, 312  
   db2\_subsystem  
     parameter 143, 310  
   delay parameter 143, 313  
   errwait parameter 143, 313  
   inamsg parameter 144, 311  
   loadxit parameter 144, 311  
   logreuse parameter 144, 311  
   logstdout parameter 145, 311  
   notify parameter 145, 312  
   operating 304  
   opt4one parameter 145, 313  
   pwdfile parameter 146, 310  
   setting up 32  
   sleep parameter 146, 312  
   spillfile parameter 147, 315

Apply program (*continued*)  
 for z/OS (*continued*)  
   starting 139, 308, 453  
   status 304  
   stopping 151, 304  
   term parameter 148, 314  
   trlreuse parameter 148, 313  
   latency analysis 167  
   messages 167  
   mini-cycles 69  
   operating 266  
   performance data 163  
   run-time processing  
     statements 112  
   scheduling 73, 463  
   spill files, storage  
     requirements 9  
   table-mode processing 72  
   throughput analysis 167  
   transaction-mode processing 72  
   user ID 21  
 Apply qualifiers  
   changing in subscription  
     sets 209  
   monitoring status 167  
   naming rules 301  
   number of associated  
     subscription sets 65  
   use when starting the Apply  
     program 139, 149  
 Apply trace (APPLYTRACE) table  
   pruning 236  
   structure 514  
 Apply trail (APPLYTRAIL) table  
   pruning 236  
   structure 515  
 APPLY\_JOB (Apply job) table 513  
   apply\_path parameter 141, 310  
   apply\_qual parameter 141, 304, 310  
 Apply-qualifier cross-reference  
   (AUTHTKN) table 484  
 APPLYTRACE (Apply trace) table  
   pruning 236  
   structure 514  
 APPLYTRAIL (Apply trail) table  
   pruning 236  
   structure 515  
 APYQUAL parameter 430  
 ARM (Automatic Restart  
   Manager) 455  
 ASCII tables 641  
 ASN messages 549  
 asnacmd command 304  
 asnanalyze command 305  
 asnappl command 308

asncap command 316  
 asncmd command 322  
 ASNDLCOPY exit routine 101  
 ASNDLCOPYD file-copy  
   daemon 105  
 ASNDONE exit routine  
   rejected transactions 56  
   using 151, 152  
 asndone.smp file 152  
 ASNL2RNx command 453  
 ASNLOAD exit routine  
   customizing behavior 157  
   description 154  
   error handling 154  
   files generated 155  
   for DATALINK replication 100  
   for OS/400 159  
   for UNIX 155  
   for Windows 155  
   for z/OS 156  
   prerequisites 154  
   using asnload.ini file 159  
   using crossloader utility 158  
 asnload.ini file 159  
 ASNMAIL exit 178  
 asnmcmd command 329  
 asnmnon command 330  
 ASNPLXFY utility 456  
 asnpwd command 335  
 asnsrct command 338  
 asnsdrop command 340  
 asntrc command 341  
 AT command  
   Apply program 463, 464  
   Capture program 463, 464  
   Replication Alert Monitor 463, 464  
 AT NetView command  
   Apply for z/OS 464  
   Capture for z/OS 464  
 attributes  
   changing for registered  
     objects 184  
   changing for subscription  
     sets 198  
 auditing  
   cold start 82  
   gap in data 82  
   source data 46  
 authentication, end-user  
   for UNIX 15, 23  
   for Windows 15, 23  
 authorization  
   for administration 17, 19  
   for Apply program 21

- authorization (*continued*)
  - for Capture program 19
  - for Capture triggers 20
  - for Replication Alert Monitor 22
- AUTHTKN (Apply-qualifier cross-reference) table 484
- automatic pruning 234
- Automatic Restart Manager (ARM) 455
- autoprune parameter
  - overview 123
  - pruning from the Replication Alert Monitor 175
  - use with asncap command 317
  - use with asncmd command 325
- autostop parameter 124, 317, 325

## B

- backup database command 27
- base aggregate tables
  - definition 78
  - structure 541
  - usage 81
- batch jobs
  - memory used by 3
  - running 453
- before-image columns
  - change-aggregate tables 92
  - registering 44
  - restrictions 46
- before-image prefix 47
- binary large object (BLOB)
  - replication considerations 98
- binding
  - Apply program
    - for UNIX 28
    - for Windows 28
    - for z/OS 32
  - Capture program
    - for UNIX 27
    - for Windows 27
    - for z/OS 32
  - Replication Alert Monitor
    - for UNIX 29
    - for Windows 29
- BLOB (binary large object)
  - replication considerations 98
- blocking factor 69

## C

- calculated columns 91
- CALL procedures
  - before and after run-time processing 112

- CALL procedures (*continued*)
  - defining for subscription set 73
- CAPCTLLIB parameter 441
- CAPENQ (Capture enqueue) table 485
- CAPMON (Capture monitor) table
  - pruning 236
  - structure 486
- CAPPARMS (Capture parameters) table
  - changing 134
  - structure 487
  - using 119
- CAPSCHEMAS (Capture schemas) table 483
- CAPSTART signals 219
- CAPSTOP signals 220
- CAPTRACE (Capture trace) table
  - pruning 236
  - structure 490
- Capture control server
  - adding to Replication Center 254
  - control tables at 483
  - multiple Capture schemas 25
- Capture control tables
  - AUTHTKN (Apply-qualifier cross-reference) 484
  - CAPENQ (Capture enqueue) 485
  - CAPMON (Capture monitor) 486
  - CAPPARMS (Capture parameters)
    - changing 134
    - structure 487
    - using 119
  - CAPSCHEMAS (Capture schemas) 483
  - CAPTRACE (Capture trace) 490
  - CCD (consistent-change-data) 491
  - CD (change-data) 493
  - list of 483
  - PRUNCNTL (pruning control) 494
  - PRUNE\_LOCK (prune lock) 496
  - PRUNE\_SET (prune set) 496
  - REG\_EXT (register extension) 497
  - REG\_SYNC (register synchronization) 505
  - REGISTER (register) 498
  - RESTART (restart) 505
  - SEQTABLE (sequencing) 507

- Capture control tables (*continued*)
  - SIGNAL (signal) 507
  - UOW (unit-of-work) 510
- Capture enqueue (CAPENQ) table 485
- Capture log file 126
- Capture monitor (CAPMON) table
  - pruning 236
  - structure 486
- Capture parameters (CAPPARMS) table
  - changing 134
  - structure 487
  - using 119
- Capture program
  - altering behavior while running 132
  - authorization requirements 19
  - changing parameter values 119
  - changing schemas 191
  - cold start prevention 237
  - commands 303
  - communicating with
    - Apply program 465, 466
    - Replication Alert Monitor 470
    - Replication Center 465
  - connectivity 15
  - for OS/400
    - authorization requirements 19
    - CAPCTLLIB parameter 441
    - changing attributes 391
    - checking status 163
    - CLNUPITV parameter 440
    - cold start parameters 438
    - cold start, automatic 445
    - creating SQL packages 30, 31
    - default parameters 118, 119
    - FRCFRQ parameter 444
    - JOB parameter 438
    - journal entry types 643
    - journals and journal receivers, managing 34
    - JRN parameter 441
    - LAG parameter 443
    - MEMLMT parameter 442
    - MONITV parameter 442
    - MONLMT parameter 442
    - operating 117
    - overriding attributes of 414
    - progress of 180
    - reinitializing 412
    - RESTART parameter 438
    - RETAIN parameter 443

## Capture program (*continued*)

### for OS/400 (*continued*)

- scheduling 464
- setting up 30, 32
- starting 132, 436
- stopping 135, 400
- TRCLMT parameter 441
- WAIT parameter 439
- warm start parameters 438

### for UNIX

- autoprune parameter 123, 317, 325
- autostop parameter 124, 317, 325
- binding 27
- capture\_path parameter 124, 317
- capture\_schema
  - parameter 125, 317, 323
- capture\_server
  - parameter 126, 317, 323
- changing parameters 322
- checking status 161
- cold start parameters 130, 320
- commit\_interval
  - parameter 126, 318, 325
- configuring 27
- default parameters 117
- lag\_limit parameter 126, 318, 325
- logreuse parameter 126, 318, 326
- logstdout parameter 127, 318, 326
- memory\_limit
  - parameter 127, 318, 326
- monitor\_interval
  - parameter 128, 318, 326
- monitor\_limit parameter 128, 318, 326
- operating 117, 322
- prune\_interval
  - parameter 128, 319, 327
- pruning 322
- reinitializing 137, 322
- resuming 136, 322
- retention\_limit
  - parameter 129, 319, 327
- setting up 26
- sleep\_interval
  - parameter 130, 319, 327
- starting 121, 316, 647
- startmode parameter 130, 320

## Capture program (*continued*)

### for UNIX (*continued*)

- status of 322
- stopping 135, 322
- suspending 136, 322
- term parameter 131, 321, 327
- trace\_limit parameter 131, 321, 328
- warm start parameters 130, 320

### for Windows

- autoprune parameter 123, 317, 325
- autostop parameter 124, 317, 325
- binding 27
- capture\_path parameter 124, 317
- capture\_schema
  - parameter 125, 317, 323
- capture\_server
  - parameter 126, 317, 323
- changing parameters 322
- checking status 161
- cold start parameters 130, 320
- commit\_interval
  - parameter 126, 318, 325
- configuring 27
- default parameters 117
- lag\_limit parameter 126, 318, 325
- logreuse parameter 126, 318, 326
- logstdout parameter 127, 318, 326
- memory\_limit
  - parameter 127, 318, 326
- monitor\_interval
  - parameter 128, 318, 326
- monitor\_limit parameter 128, 318, 326
- operating 117, 322
- prune\_interval
  - parameter 128, 319, 327
- pruning 322
- reinitializing 137, 322
- resuming 136, 322
- retention\_limit
  - parameter 129, 319, 327
- setting up 26
- sleep\_interval
  - parameter 130, 319, 327
- starting 121, 316, 647

## Capture program (*continued*)

### for Windows (*continued*)

- startmode parameter 130, 320
- status of 322
- stopping 135, 322
- suspending 136, 322
- term parameter 131, 321, 327
- trace\_limit parameter 131, 321, 328
- warm start parameters 130, 320

### for z/OS

- autoprune parameter 123, 317, 325
- autostop parameter 124, 317, 325
- capture\_path parameter 124, 317
- capture\_schema
  - parameter 125, 317, 323
- capture\_server
  - parameter 126, 317, 323
- changing parameters 322
- checking status 161
- cold start parameters 130, 320
- commit\_interval
  - parameter 126, 318, 325
- default parameters 117
- lag\_limit parameter 126, 318, 325
- logreuse parameter 126, 318, 326
- logstdout parameter 127
- memory\_limit
  - parameter 127, 318, 326
- monitor\_interval
  - parameter 128, 318, 326
- monitor\_limit parameter 128, 318, 326
- operating 117, 322
- prune\_interval
  - parameter 128, 319, 327
- pruning 322
- reinitializing 137, 322
- resuming 136, 322
- retention\_limit
  - parameter 129, 319, 327
- setting up 32
- sleep\_interval
  - parameter 130, 319, 327
- starting 121, 316, 453
- startmode parameter 130, 320

- Capture program (*continued*)
  - for z/OS (*continued*)
    - status of 322
    - stopping 135, 322
    - suspending 136, 322
    - term parameter 131, 321, 327
    - trace\_limit parameter 131, 321, 328
    - warm start parameters 130, 320
  - latency analysis 166
  - memory used by 3
  - messages 165
  - operating 265
  - performance data 163
  - running more than one 25
  - scheduling 463
  - setting defaults for parameters 119
  - setting environment variables 26
  - signals 214
  - throughput analysis 165
  - user ID 19
  - where to start it 126
- Capture schemas
  - changing 191
  - naming rules 301
  - using multiple 25
- Capture schemas (CAPSCHEMAS) table 483
- Capture signals 214
- Capture trace (CAPTRACE) table
  - pruning 236
  - structure 490
- Capture triggers
  - authorization requirements 20
  - communicating with
    - Apply program 465, 468
    - Replication Center 465
  - conflicts with pre-existing triggers 13
  - names of 13
  - planning 12
- capture\_path parameter 124, 317
- capture\_schema parameter 125, 317, 323
- capture\_server parameter 126, 317, 323
- catalog tables, registering 37
- CCD (consistent-change-data) tables
  - adding UOW columns 83
  - external
    - multi-tier replication 85
- CCD (consistent-change-data) tables (*continued*)
  - internal
    - multiple targets 83
  - locks on 12
  - non-DB2 relational data sources
    - using CCD tables 39
  - nonrelational data sources
    - maintaining CCD tables 61
    - using CCD tables 37
  - replication sources 85
  - structure
    - Capture control server 491
    - target server 543
  - usage
    - history or audit 82
    - multi-tier replication 85
- CD (change-data) tables
  - for joins 58
  - for views 58
  - pruning 235
  - storage requirements 9
  - structure 493
  - summarizing contents 82
  - triggers on 108
- CD (change-data) views 58
- change aggregate tables
  - definition 78
  - structure 542
  - usage 82
- change capture
  - enabling 255
- change-capture replication
  - description 43
  - registration option 42
- change-data (CD) tables
  - pruning 235
  - storage requirements 9
  - structure 493
  - summarizing contents 82
- changing Capture parameters
  - for OS/400 391
  - for UNIX 322
  - for Windows 322
  - for z/OS 322
- character large object (CLOB)
  - replication considerations 98
- CHGDPRCAPA command 391
- CHGJRN command 35
- CLNUPITV parameter 440
- CLOB (character large object)
  - replication considerations 98
- code pages
  - compatible 13
- code pages (*continued*)
  - DB2CODEPAGE environment variable 14
  - translation 13
- cold start, Capture program
  - for OS/400 438, 445
  - for UNIX 130, 320
  - for Windows 130, 320
  - for z/OS 130, 320
  - preventing 237
- cold startmode 131
- column (vertical) subsetting
  - at the source 42
  - at the target 91
- columns
  - adding to registered source tables 185
  - after-image 44
  - available for replication 42
  - before-image 44
  - calculated 91
  - computed 113
  - defining in target table 91
  - mapping from sources to targets 92
  - registering in source table 42
  - relative record numbers on OS/400 57
  - renaming 113
  - subsetting
    - at the source 42
    - at the target 91
- commands
  - See replication commands
- commit\_interval parameter
  - overview 126
  - tuning 4
  - use with asncap command 318
  - use with asncmd command 325
- compression dictionaries (z/OS) 229
- computed columns
  - CD table 82
  - creating 113
  - source table 81
- CONDITIONS (Monitor conditions) table 534
- configuration parameters for DB2
  - APPLHEAPSZ 27
  - DBHEAP 27
  - LOGBUFSZ 27
  - LOGFILSIZ 27
  - LOGPRIMARY 27
  - LOGSECOND 27

- configuration parameters for DB2  
(*continued*)
  - MAXAPPLS 27
- configuring
  - Apply program
    - for UNIX 28
    - for Windows 28
  - Capture program
    - for UNIX 27
    - for Windows 27
  - connectivity 15
  - Replication Alert Monitor
    - for UNIX 29
    - for Windows 29
- conflict detection
  - levels of 55
  - overview 54
  - peer-to-peer replication 11
  - planning 11
  - requirements 46
  - update-anywhere replication 11
- conflicts
  - preventing 11
- connecting
  - to iSeries server 16
  - to z/OS server 16
- connectivity
  - between DB2 platforms 15, 16
  - failure recovery for control tables 238
- consistent-change-data (CCD) tables
  - adding UOW columns 83
  - external
    - multi-tier replication 85
  - internal
    - multiple targets 83
  - locks on 12
  - non-DB2 relational data sources
    - using CCD tables 39
  - nonrelational data sources
    - maintaining CCD tables 61
    - using CCD tables 37
  - replication sources 85
  - structure
    - Capture control server 491
    - target server 543
  - usage
    - history or audit 82
    - multi-tier replication 85
- CONTACTGRP (Monitor group contacts) table 537
- contacts
  - for Replication Alert Monitor 170
- CONTACTS (Monitor contacts) table 537
- control servers, adding to Replication Center 254
- control tables
  - ALERTS (Monitor alerts) 533
  - APPENQ (Apply enqueue) 513
  - Apply
    - creating 253
  - Apply control server 479
  - APPLY\_JOB (Apply job) 513
  - APPLYTRACE (Apply trace) 514
  - APPLYTRAIL (Apply trail) 515
  - at Apply control server 513
  - at Capture control server 483
  - at Monitor control server 533
  - authorization requirements for OS/400 32
  - AUTHTKN (Apply-qualifier cross-reference) 484
  - CAPENQ (Capture enqueue) 485
  - CAPMON (Capture monitor)
    - pruning 236
    - structure 486
  - CAPPARMS (Capture parameters)
    - structure 487
  - CAPSCHEMAS (Capture schemas) 483
  - CAPTRACE (Capture trace)
    - pruning 236
    - structure 490
  - Capture
    - creating 252
  - Capture server 476
  - CCD (consistent-change-data)
    - Capture control server 491
    - target server 543
  - CD (change-data) 493
  - CONDITIONS (Monitor conditions) 534
  - connectivity failure recovery 238
  - CONTACTGRP (Monitor group contacts) 537
  - CONTACTS (Monitor contacts) 537
  - creating
    - for Apply 253
    - for Capture 252
    - for non-DB2 relational sources 24
    - for Replication Alert Monitor 169, 253
- control tables (*continued*)
  - creating (*continued*)
    - multiple sets 25
    - on OS/400 24, 396
    - on UNIX, Windows 23
    - on z/OS 24
  - dropping
    - for Replication Alert Monitor 170
  - dynamic 231
  - granting authority for OS/400 19, 403
  - GROUPS (Monitor groups) 538
  - I/O error recovery 238
  - maintaining 231
  - MONENQ (Monitor enqueue) 538
  - Monitor
    - creating 253
  - MONSERVERS (Monitor servers) 539
  - MONTRACE (Monitor trace) 540
  - MONTRAIL (Monitor trail) 540
  - profiles 249
  - PRUNCNTL (pruning control) 494
  - PRUNE\_LOCK (prune lock) 496
  - PRUNE\_SET (prune set) 496
  - pruning 233
  - quick reference
    - Apply control server 479
    - at a glance 472
    - Capture server 476
    - target server 482
  - rebinding, packages and plans 232
  - REG\_EXT (register extension) 497
  - REG\_SYNCH (register synchronization) 505
  - REGISTER (register) 498
  - reorganizing 232
  - RESTART (restart) 505
  - revoking authority for OS/400 425
  - RUNSTATS utility 231
  - SEQTABLE (sequencing) 507
  - SIGNAL (signal) 507
  - static 233
  - storage requirements 8
  - SUBS\_COLS (subscription columns) 519
  - SUBS\_EVENT (subscription events) 521



- control tables (*continued*)
  - SUBS\_MEMBR (subscription members) 522
  - SUBS\_SET (subscription sets) 526
  - SUBS\_STMTS (subscription statements) 531
  - target server 482
  - UOW (unit-of-work) 510
- control\_server parameter 142, 304, 310
- copying replication
  - configurations 221
- copyonce parameter 142, 312
- COPYONCE parameter 433
- correlation ID 58
- creating control tables 23
- creating subscription sets 257
- crossloader utility 158
- CRTDPRTBL command 396
- CRTJRN command 33
- CRTJRNRCV command 33
- CTLSVR parameter 430
- current receiver size 7, 34
- customizing, SQL scripts 115

## D

- data
  - advanced subsetting
    - techniques 107
  - displaying historical 163
  - manipulating 111
  - preventing double-deletes 60
  - retrieving from source
    - tables 239
  - subsetting
    - during registration 107
    - using predicates 109
    - using triggers on CD
      - tables 108
    - using views 108
    - using views to specify
      - predicates 109
  - transforming
    - at registration 111
    - at subscription 112
    - creating computed
      - columns 113
    - renaming columns 113
- data blocking 69
- data consistency 89
- data encryption restrictions 97
- Data Links
  - replication 99
- Data Links Manager replication
  - daemon 103
- data types
  - mapping between columns 92
  - replicating
    - DATALINK values 99
    - large objects (LOB) 98
    - restrictions 97
- data-sharing mode 456
- databases, enabling for change
  - capture 255
- DATALINK values
  - ASNDLCOPY exit routine 101
  - ASNDLCOPYD file-copy
    - daemon 105
  - DLFM\_ASNCOPYD file-copy
    - daemon 103
  - replicating 99
  - restrictions 54, 87
  - storing updates 49
- DB2 Enterprise Server Edition
  - restrictions 42
- DB2 Extenders
  - restrictions 98
- DB2 replication
  - authorization requirements 17
- DB2 tables
  - registering 37
- DB2 views
  - registering 60
- db2\_subsystem parameter 143, 310
- DB2CODEPAGE environment
  - variable 14, 27
- DB2DBDFT environment
  - variable 27
- DB2INSTANCE environment
  - variable 26
- db2rc command 245
- DBADM 18, 19
- DBCLOB (double-byte character
  - large object)
    - replication considerations 98
- DBHEAP configuration
  - parameter 27
- deactivating
  - registered objects 188
  - subscription sets 68, 212
- deactivating subscription sets 261
- defaults
  - for Apply parameters
    - (OS/400) 150
  - for Apply parameters (UNIX,
    - Windows, z/OS) 140
  - for Capture parameters
    - (OS/400) 118, 119

- defaults (*continued*)
  - for Capture parameters (UNIX,
    - Windows, z/OS) 117, 123
- delay parameter 143, 313
- DELAY parameter 433
- delete journal receiver exit routine
  - about 35
  - registering 36
  - removing 36
- delimiters, in generated SQL
  - scripts 116
- diagnostic files
  - storage 9, 10
- differential refresh replication
  - See change-capture replication
- disk space
  - requirements 5
  - temporary files 9
- display names 460
- distinct data types 97
- distributed recovery points 217
- DLFM\_ASNCOPYD file-copy
  - daemon 103
- double-byte character large object
  - (DBCLOB)
    - replication considerations 98
- double-deletes 60
- DPR registrations (OS/400)
  - adding 349
  - removing 420
- DSPIRN command 180
- dynamic control tables 231

## E

- editing, SQL scripts 115
- EDITPROC clauses
  - restrictions, compression 97
- email\_server parameter 176
- ENDDPRAPY command 397
- ENDDPRCAP command 135, 400
- ENDJOB command 401
- environment variables
  - Capture program 26
  - DB2CODEPAGE 14, 27
  - DB2DBDFT 27
  - DB2INSTANCE 26
  - LIBPATH 27
- errwait parameter 143, 313
- event-based scheduling 74
- events, coordinating 214
- existing tables as targets 89
- exit routines
  - ASNDLCOPY 101
  - ASNDONE
    - using 151, 152

exit routines (*continued*)

ASNLOAD

- customizing 157
- for OS/400 159
- for UNIX 155
- for Windows 155
- for z/OS 156
- using 154

delete journal receiver  
(OS/400) 35

external CCD tables

multi-tier replication 85

## F

FIELDPROC clauses

restrictions, compression 97

file-copy daemons

ASNDLCOPYD 105

DLFM\_ASNCOPYD 103

files

- \*.APP.log 144
- \*.CAP.log 126
- \*.err 147
- \*.sqs 147, 148
- asndone.smp 152
- asnload.ini 159
- spill 9

fragmentation

horizontal

- at the source 43
- at the target 91

peer-to-peer replication 11

update-anywhere replication 11

vertical

- at the source 42
- at the target 91

FRCFRQ parameter 444

full-refresh copying

Apply for iSeries 57, 431

forcing 263

registration option 42

FULLREFPGM parameter 431

## G

gap detection 82

generated SQL scripts 115

global record 499

GROUPS (Monitor groups)

table 538

GRTPRAUT command 31

syntax 403

GRTOBJAUT command 31

## H

heterogeneous replication

registering sources 39

heterogeneous replication (*continued*)

restrictions

aggregate tables 81

CCD tables 44

multi-tier replication 85

update-anywhere 49, 87

history data

CCD tables 82

source data 46

horizontal (row) subsetting

at the source 43

at the target 91

## I

I/O error recovery, control

tables 238

IMS data sources

maintaining CCD tables 61

registering 37

using CCD tables 37

IMS DataPropagator 37

inactive subscription sets 68

INACTMSG parameter 432

inamsg parameter 144, 311

indexes

target tables 93

inner-joins as sources 58

internal CCD tables

multiple targets 83

interval timing 73

invocation parameters

Analyzer

for OS/400 388

for UNIX 306

for Windows 306

Apply program

for OS/400 149, 429

for UNIX 140, 309

for Windows 140, 309

for z/OS 140, 309

Capture program

for OS/400 117, 132, 438

for UNIX 123, 317

for Windows 123, 317

for z/OS 123, 317

Replication Alert Monitor

for UNIX 331

for Windows 331

for z/OS 331

INZDPRCAP command 412

iSeries server

connecting to 16

## J

JCL

starting the Apply program 453

starting the Capture

program 453

starting the Replication Alert

Monitor 453

JOBD parameter 429, 438

JOIN\_UOW\_CD column 109

joins as sources 58

journal jobs

checking status 163

journal message queues 35

journal receivers

creating for source tables 33

current, size 7

delete journal receiver exit

routine 35

maintaining 226

managing 34

retaining 229

system management 34

threshold 34

user management 35

journals

creating 33

creating for source tables 32

default message queue 35

entry types 643

managing 34

QSQRJRN journal 32

registering as sources 37

setup 32

starting 33

using 32

using remote journal

function 56

JRN parameter 441

## L

LAG parameter 443

lag\_limit parameter 126, 318, 325

large object (LOB)

replication considerations 98

large replication jobs 69

latency

Apply program 167

Capture program 166

launchpad 246

LIBPATH 27

loadxit parameter 144, 311

LOB (large object)

replication considerations 98

update-anywhere restrictions 87

- locks
  - on CCD tables 12
- log
  - planning impact to 12
- log records
  - archived before captured 7
  - compression dictionaries (z/OS) 229
  - maintaining 226
  - retaining 226
- LOGBUFSZ configuration
  - parameter 27
- LOGFILSIZ configuration
  - parameter 27
- logging requirements
  - DB2 source servers 6
  - non-DB2 relational source servers 12
  - target servers 7
- logical partitioning keys
  - description 48
- LOGPRIMARY configuration
  - parameter 27
- logreuse parameter (for Apply) 144, 311
- logreuse parameter (for Capture) 126, 318, 326
- logreuse parameter (for Replication Alert Monitor) 174
- LOGSECOND configuration
  - parameter 27
- logstdout parameter (for Apply) 145, 311
- logstdout parameter (for Capture) 127, 318, 326
- logstdout parameter (for Replication Alert Monitor) 174
- LONG VARGRAPHIC data
  - types 97

## M

- manipulating data
  - at registration 111
  - at subscription 112
  - creating computed columns 113
  - renaming columns 113
- mapping
  - data types between tables 92
  - source columns to target columns 92
  - sources to targets 75
- master tables (update-anywhere)
  - overview 87
  - recapturing changes 49
- max\_notifications\_minutes
  - parameter 176
- max\_notifications\_per\_alert
  - parameter 176
- MAX\_SYNCH\_MINUTES, data
  - blocking 69
- MAXAPPLS configuration
  - parameter 27
- MEMLMT parameter 442
- memory
  - Apply program 5
  - batch jobs 3
  - Capture program 3
  - planning 3
  - reading log records 4
  - registrations 4
  - Replication Alert Monitor 5
  - subscription sets 5
  - transactions 3
  - using CAPMON table to tune 4
- memory\_limit parameter
  - overview 127
  - tuning 4
  - use with asncap command 318
  - use with asncmd command 326
- merging
  - subscription sets 206
  - triggers 13
- message queues, for journals 35
- messages 165, 167, 549
- Microsoft SQL Server
  - replication restrictions 45
- mini-cycles 69
- MODIFY command 454
- MONENQ (Monitor enqueue)
  - table 538
- Monitor
  - See Replication Alert Monitor
  - Monitor alerts (ALERTS) table 533
  - Monitor conditions (CONDITIONS) table 534
  - Monitor contacts (CONTACTS) table 537
  - Monitor control server
    - adding to Replication Center 254
    - control tables at 533
    - creating control tables 169
  - Monitor control tables
    - ALERTS (Monitor alerts) 533
    - CONDITIONS (Monitor conditions) 534
    - CONTACTGRP (Monitor group contacts) 537

- Monitor control tables (*continued*)
  - CONTACTS (Monitor contacts) 537
  - GROUPS (Monitor groups) 538
  - list of 533
  - MONENQ (Monitor enqueue) 538
  - MONSERVERS (Monitor servers) 539
  - MONTRACE (Monitor trace) 540
  - MONTRAIL (Monitor trail) 540
- Monitor enqueue (MONENQ)
  - table 538
- Monitor group contacts
  - (CONTACTGRP) table 537
- Monitor groups (GROUPS)
  - table 538
- Monitor qualifiers, naming
  - rules 301
- Monitor servers (MONSERVERS)
  - table 539
- Monitor trace (MONTRACE)
  - table 540
- Monitor trail (MONTRAIL)
  - table 540
- monitor\_errors parameter 178
- monitor\_interval parameter (for Capture) 128, 318, 326
- monitor\_interval parameter (for Replication Alert Monitor) 174
- monitor\_limit parameter 128, 318, 326
- monitor\_path parameter 174
- monitoring
  - automated 168
  - for OS/400 180
  - historical trends 163
  - status of programs 163
- MONITV parameter 442
- MONLMT parameter 442
- MONSERVERS (Monitor servers)
  - table 539
- MONTRACE (Monitor trace)
  - table 540
- MONTRAIL (Monitor trail)
  - table 540
- multi-tier replication
  - defining subscription sets 85
- multiple target tables 83
- MVS console 453

## N

- names
  - Apply qualifier rules 301

## names (*continued*)

- Capture schema rules 301
- display names 460
- for Windows services 302
- Monitor qualifier rules 301
- of Capture triggers 13
- of replication services 460
- subscription sets 199
- national language support (NLS) 14
- network connectivity 15
- nicknames
  - for crossloader utility 158
  - registering 39
  - restrictions
    - aggregate tables 81
    - multi-tier replication 85
    - update-anywhere 49, 87
    - with CCD tables 44
- NLS (national language support) 14
- non-DB2 relational data sources
  - locks 12
  - registering 39
  - restrictions
    - aggregate tables 81
    - multi-tier replication 85
    - update-anywhere 49, 54, 87
  - source servers 12
  - using CCD tables 39
- nonrelational data sources
  - maintaining CCD tables 61
  - using CCD tables 37
- notify parameter 145, 312

## O

### objects

- changing attributes 184
- deactivating 188
- reactivating 189
- registering 183
- stop capturing changes 188

### operating

- Apply program 266, 304
- Capture program 265, 322
- Replication Alert Monitor 266, 329

opt4one parameter 145, 313

OPTSNGSET parameter 434

### Oracle sources

- restrictions 45

OS/400 data sources

- with remote journaling 56

### overriding attributes (OS/400)

- Capture program 414

OVRDPRCAPA command 414

## P

packages, rebinding 232

### parameters, invocation

#### Analyzer

- for OS/400 388
- for UNIX 306
- for Windows 306

#### Apply program

- for OS/400 149, 429
- for UNIX 140, 309
- for Windows 140, 309
- for z/OS 140, 309

#### Capture program

- for OS/400 438
- for UNIX 123, 317
- for Windows 123, 317
- for z/OS 123, 317

#### Replication Alert Monitor

- for UNIX 331
- for Windows 331
- for z/OS 331

### password files

- asnpwd command 335
- storing 23

### passwords for Replication

Center 247

### peer-to-peer replication, conflict

detection 11

### planning

- coexistence of triggers 13
- conflict detection 11, 54
- locks on CCD tables 12
- log impact 6, 12
- memory 3
- storage requirements 5
- transaction throughput rates 12

### plans, rebinding 232

### point-in-time tables

- structure 546
- usage 81

### predicates

- defining for target tables 91
- subsetting 109

PREDICATES column 109

prefix, before-image 47

### primary keys

- logical partitioning 48
- relative record numbers for OS/400 57
- used as target key 94

### profiles

- control-table 249
- description 248
- source-object 250
- target-object 250

### promoting

- registered tables or views 262
- replication configurations 221
- subscription sets 263

### PRUNCNTL (pruning control)

table 494

### prune lock (PRUNE\_LOCK)

table 496

prune set (PRUNE\_SET) table 496

prune\_interval parameter 128, 319, 327

### PRUNE\_LOCK (prune lock)

table 496

PRUNE\_SET (prune set) table 496

### pruning

Apply trace (APPLYTRACE)  
table 236

Apply trail (APPLYTRAIL)  
table 236

Capture monitor (CAPMON)  
table 236

Capture program  
for UNIX 322  
for Windows 322  
for z/OS 322

Capture trace (CAPTRACE)  
table 236

CD (change-data) tables 235

control tables 233

signal (SIGNAL) table 236

UOW (unit-of-work) table 235, 510

pruning control (PRUNCNTL)  
table 494

pwdfile parameter 146, 310

## R

RCVIRNE command 34

### reactivating

- objects 189
- registrations 189
- tables 189

read dependencies 56

rebinding, packages and plans 232

### recapturing changes

(update-anywhere) 49

receiver size, current 7

recovery points, distributed 217

referential integrity 89

REG\_EXT (register extension)  
table 497

REG\_SYNCH (register  
synchronization) table 505

register (REGISTER) table 498

REGISTER (register) table 498

- register extension (REG\_EXT)
  - table 497
- register synchronization (REG\_SYNCH) table 505
- registering
  - DB2 tables 37
  - IMS data sources 37
  - non-DB2 relational data
    - sources 39
  - objects 183
  - options for sources
    - after-image columns 44
    - before-image columns 44
    - before-image prefix 47
    - change-capture
      - replication 42
    - column (vertical)
      - subsetting 42
    - conflict detection 54
    - full-refresh copying 42
    - recapturing changes
      - (update-anywhere) 49
    - relative record numbers 57
    - row (horizontal)
      - subsetting 43
    - stop Capture on error 48
    - updates as deletes and inserts 48
    - using remote journals 56
  - tables 183
  - views
    - overview 58, 60
    - procedure 183
- registering sources 256
- registrations
  - adding 349
  - adding columns 185
  - attributes, changing 184
  - deactivating 188
  - reactivating 189
  - removing 190, 420
  - stop capturing changes 188
- registry variables
  - DB2CODEPAGE 14, 27
  - DB2DBDFT 27
  - DB2INSTANCE 26
- reinitializing Capture program
  - for UNIX 137
  - for Windows 137
  - for z/OS 137
- relative record numbers
  - as primary key for OS/400 57
  - support for OS/400 57
  - used as target key 94
- relative timing 73
- remote journals as sources 56
- remote source tables 56
- renaming columns 113
- reorganizing
  - control tables 232
- replica tables
  - defining read-write targets 87
  - definition 78
  - recapturing changes 49
  - structure 546
- Replication Alert Monitor
  - about 168
  - ASNMAIL exit 178
  - authorization requirements 22
  - changing contacts 170
  - communicating with
    - Apply program 470
    - Capture 470
    - Replication Center 469
  - control tables 169
  - copying contacts 171
  - defining contacts 170
  - for UNIX
    - binding 29
    - checking status 161
    - operating 329
    - starting 173, 330, 647
  - for Windows
    - binding 29
    - checking status 161
    - operating 329
    - starting 173, 330, 647
  - for z/OS
    - checking status 161
    - operating 329
    - starting 173, 330, 453
  - memory usage 5
  - operating 266
  - pruning 175
  - reinitializing 179
  - scheduling 178, 463, 464
  - selecting alert conditions 171
  - setting notification 176, 178
  - specifying run times 174
  - stopping 179
  - storing output from 174
  - tracing 175
- Replication Analyzer
  - for OS/400
    - creating SQL packages 31
    - invocation parameters 388
  - for UNIX, invocation
    - parameters 306
  - for Windows, invocation
    - parameters 306
- Replication Center
  - activating subscription sets 261
  - adding servers 254
  - communicating with
    - Apply program 465
    - Capture program 465
    - Capture triggers 465
    - Replication Alert
      - Monitor 469
  - connectivity 15
  - control tables 251
  - control-table profiles 249
  - creating subscription sets 257
  - deactivating subscription
    - sets 261
  - deleting definitions 264
  - description 243
  - enabling databases for change
    - capture 255
  - forcing full refresh 263
  - launchpad 246
  - operating Apply program 266
  - operating Capture program 265
  - operating Replication Alert
    - Monitor 266
  - profiles 248
  - promote functions 221
  - promoting registered tables or
    - views 262
  - promoting subscription sets 263
  - registering sources 256
  - removing definitions 264
  - source-object profiles 250
  - starting 245
  - target-object profiles 250
  - user IDs and passwords 247
- replication commands
  - STA JES2
    - Apply for z/OS 464
    - Capture for z/OS 464
  - ADDJOBSCDE 464
  - ASNL2RNx 453
  - AT 463, 464
  - AT NetView
    - Apply for z/OS 464
    - Capture for z/OS 464
  - backup database 27
  - CRTJRNRCV 33
  - db2rc 245
  - DSPJRN 180
  - for OS/400
    - ADDDPRREG 349
    - ADDDPRSUB 359
    - ADDDPRSUBM 376
    - ADDEXITPGM 36

- replication commands *(continued)*
  - for OS/400 *(continued)*
    - ANZDPR 387
    - ANZDPRJRN 35
    - CHGDPRCAP 391
    - CHGJRN 35
    - CRTDPRTBL 396
    - CRTJRN 33
    - ENDDPRAPY 397
    - ENDDPRCAP 135, 400
    - ENDJOB 401
    - GRDPRAUT 31, 403
    - GRTOBJAUT 31
    - INZDPRCAP 412
    - OVRDPRCAP 414
    - RCVJRNE 34
    - RMVDPRREG 420
    - RMVDPRSUB 421
    - RMVDPRSUBM 423
    - RMVEXITPGM 36
    - RVKDPAUT 425
    - SBMJOB 464
    - STRDPAPY 150, 427
    - STRDPRCAP 436
    - STRJRNPF 33
    - WRKDPRTRC 446
    - WRKJOB 163
    - WRKREGINF 36
    - WRKSBJOB 163
    - WRKSBSJOB 163
  - for UNIX
    - asnacmd 304
    - asnanalyze 305
    - asnapply 308
    - asncap 316
    - asnccmd 322
    - asnmcmd 329
    - asnmon 330
    - asnpwd 335
    - asntrc 341
  - for Windows
    - asnacmd 304
    - asnanalyze 305
    - asnapply 308
    - asncap 316
    - asnccmd 322
    - asnmcmd 329
    - asnmon 330
    - asnpwd 335
    - asnscrt 338
    - asnsdrop 340
    - asntrc 341
  - for z/OS
    - asnacmd 304
    - asnapply 308

- replication commands *(continued)*
  - for z/OS *(continued)*
    - asncap 316
    - asnccmd 322
    - asnmcmd 329
    - asnmon 330
    - asntrc 341
    - MODIFY 454
    - update database
      - configuration 27
  - replication environments
    - copying 221
  - replication events coordination 214
  - replication messages 549
  - replication services
    - creating 338, 459
    - dropping 340, 461
    - names 460
    - operating 461
  - replication sources
    - CCD (consistent-change-data)
      - tables 85
    - joins 58
    - maintaining CCD tables 61
    - mapping to targets 75
    - registering
      - columns 42
      - DB2 tables 37
      - IMS data sources 37
      - non-DB2 relational data
        - sources 39
      - rows 43
      - views 60
    - subscribing to 66
  - restart (RESTART) table 505
  - RESTART (restart) table 505
  - RESTART parameter 438
  - restrictions
    - abstract data types 97
    - ASCII tables 641
    - CCD tables 87
    - column names, limits 46
    - data encryption 97
    - data types 97
    - DATALINK values 54, 87
    - DB2 Enterprise Server
      - Edition 42
    - DB2 Extenders large objects 98
    - distinct data types 97
    - EDITPROC clauses 97
    - existing target tables 90
    - FIELDPROC clauses 97
    - heterogeneous replication 45, 85, 87
    - LOB data types 87

- restrictions *(continued)*
  - LONG VARGRAPHIC data
    - types 97
  - Microsoft SQL Server 45
  - non-DB2 relational data
    - sources 49, 54
  - Oracle sources 45
  - spatial data types 97
  - stored procedures 112
  - Sybase 45
  - Unicode tables 641
  - user-defined data types 97
  - VALIDPROC clauses 97
  - views 60
  - WHERE clause 92
- resuming
  - Capture program
    - for UNIX 136, 322
    - for Windows 136, 322
    - for z/OS 136, 322
- RETAIN parameter 443
- retention\_limit parameter 129, 319, 327
- RMVDPRREG command 420
- RMVDPRSUB command 421
- RMVDPRSUBM command 423
- RMVEXITPGM command 36
- roll-forward recovery 27
- row (horizontal) subsetting
  - at the source 43
  - at the target 91
- row-capture rules 43
- ROWID 98
- rows
  - available for replication 43
  - defining in target table 91
  - registering in source table 43
  - subsetting
    - at the source 43
    - at the target 91
- RRN 57
- RTYWAIT parameter 433
- run-time processing 73, 112
- running, SQL scripts 115
- runonce parameter 174
- RUNSTATS utility 231
- RVKDPAUT command 425

**S**

- SBMJOB command 464
- scenario
  - create Apply control tables 277
  - create Apply password file 283
  - create Capture control
    - tables 272

- scenario *(continued)*
  - create contacts 293
  - create Monitor control tables 291
  - create subscription set 277
  - enable source database for replication 273
  - monitoring replication 290
  - operations 286
  - planning 270
  - prerequisites 269
  - register a source 274
  - replicate data 284
  - select alert conditions for Apply program 294
  - select alert conditions for Capture program 293
  - setup 272
  - start the Replication Alert Monitor 296
  - status for Apply program 288
  - status for Capture program 287
  - stop Capture and Apply programs 289
  - update source table 286
- scheduling
  - replication programs 463
  - subscription sets 73, 74
- schemas
  - changing 191
  - naming rules 301
- SCM (Service Control Manager)
  - creating replication services 338, 459
  - dropping replication services 340, 461
  - naming replication services 460
  - operating replication services 461
- SEQTABLE (sequencing) table 507
- sequencing (SEQTABLE) table 507
- servers
  - adding to Replication Center 254
- Service Control Manager (SCM)
  - creating replication services 338, 459
  - dropping replication services 340, 461
  - naming replication services 460
  - operating replication services 461
- services
  - Windows SCM 459
- setting environment variables
  - Capture program 26
- setting up
  - Apply programs
    - for OS/400 30
    - for UNIX 26
    - for Windows 26
  - Capture programs
    - for OS/400 30
    - for UNIX 26
    - for Windows 26
  - journals 32
  - Replication Alert Monitor 29
- signal (SIGNAL) table
  - pruning 236
  - structure 507
- SIGNAL (signal) table
  - pruning 236
  - structure 507
- signals
  - CAPSTART 219
  - CAPSTOP 220
  - setting distributed recovery points 217
  - STOP 216, 217
  - USER 214
- sleep parameter 146, 312
- sleep\_interval parameter 130, 319, 327
- source logs, maintaining 226
- source servers
  - DB2
    - log impact 6
    - non-DB2 relational log impact 12
- source systems, maintaining 225
- source tables
  - adding columns 185
  - creating journals for 32
  - maintaining 225
  - retrieving lost data 239
- sources
  - CCD (consistent-change-data) tables 85
  - maintaining CCD tables 61
  - mapping to targets 75
  - profiles 250
  - promoting 262
  - registering
    - DB2 tables 37
    - IMS data sources 37
    - non-DB2 relational 39
    - Replication Center 256
    - views 58, 60
  - registering columns 42
- sources *(continued)*
  - registering rows 43
  - registration options
    - after-image columns 44
    - before-image columns 44
    - before-image prefix 47
    - change-capture replication 42
    - column (vertical) subsetting 42
    - conflict detection 54
    - full-refresh copying 42
    - recapturing changes (update-anywhere) 49
    - relative record numbers 57
    - row (horizontal) subsetting 43
    - stop Capture on error 48
    - updates as deletes and inserts 48
    - using remote journals 56
  - subscribing to 66
  - spatial data types 97
  - special data types
    - replicating
      - DATALINK values 99
      - large objects (LOB) 98
  - spill files
    - storage for Apply 10
    - storage for Capture 10
    - storage for diagnostic files 9
  - spillfile parameter 147, 315
  - splitting
    - subscription sets 201
  - SQL files, editing 115
  - SQL packages
    - creating for Apply program 31
    - creating for Capture program 30, 31
    - creating for Replication Analyzer 31
  - SQL scripts 115
  - SQL statements
    - defining for subscription set 73
    - run-time processing 112
  - sqlerrorcontinue parameter 147, 314
  - staged replication 85
  - staging data 85
  - starting
    - Apply program
      - for OS/400 149, 427
      - for UNIX 139, 308, 647
      - for Windows 139, 308, 647
      - for z/OS 139, 308, 453

- starting (*continued*)
  - Capture program
    - for OS/400 132, 436
    - for UNIX 121, 316, 647
    - for Windows 121, 316, 647
    - for z/OS 121, 316, 453
    - using Windows services 459
  - Replication Alert Monitor
    - for UNIX 173, 330, 647
    - for Windows 173, 330, 647
    - for z/OS 173, 330, 453
- starting Replication Center 245
- startmode parameter 130, 320
- static control tables 233
- status
  - Apply program 161, 163
  - Capture program 161, 163
  - journal jobs 163
  - Replication Alert Monitor 161
- stop Capture on error option 48
- stop capturing changes 188
- STOP signals 216, 217
- stopping
  - Apply program
    - for OS/400 151, 397
    - for UNIX 151, 304
    - for Windows 151, 304
    - for z/OS 151, 304
  - Capture program
    - for OS/400 135, 400
    - for UNIX 135, 322
    - for Windows 135, 322
    - for z/OS 135, 322
  - Replication Alert Monitor
    - for UNIX 179, 329
    - for Windows 179, 329
    - for z/OS 179, 329
- storage
  - Apply diagnostic files 10
  - Apply spill files 10
  - Capture diagnostic files 10
  - Capture spill files 10
  - CD table 9
  - control tables 8
  - database log and journal data 6
  - diagnostic files 9
  - requirements 5
  - target tables 8
  - temporary files 9
  - UOW table 9
- stored procedures
  - defining for subscription set 73
  - manipulating data 112
- STRDPAPY command 150, 427
- STRDPRCAP command 436
- STRJRNP command 33
- SUBNFYPGM parameter 432
- SUBS\_COLS (subscription columns) table 519
- SUBS\_EVENT (subscription events) table
  - posting events 74
  - structure 521
- SUBS\_MEMBR (subscription members) table 158, 522
- SUBS\_SET (subscription sets) table 526
- SUBS\_STMTS (subscription statements) table 531
- subscribing to sources 66
- subscription columns (SUBS\_COLS) table 519
- subscription cycle 69
- subscription events (SUBS\_EVENT) table
  - posting events 74
  - structure 521
- subscription members (SUBS\_MEMBR) table 158, 522
- subscription sets
  - activating 261
  - activation level 68
  - adding 359
  - adding members 75, 195
  - changing
    - Apply qualifiers 209
    - attributes 198
    - names 199
  - columns 91
  - creating 66, 194, 257
  - data consistency 89
  - deactivating 212, 261
  - merging 206
  - mini-cycles 69
  - multi-tier replication 85
  - number of Apply qualifiers 65
  - processing mode 72
  - promoting 263
  - referential integrity 89
  - removing 213, 421
  - rows 91
  - run-time processing statements 112
  - scheduling
    - event-based 74
    - time-based 73
  - splitting 201
  - SQL statements 73
  - stored procedures 73
  - update-anywhere replication 87
- subscription sets (SUBS\_SET) table 526
- subscription statements (SUBS\_STMTS) table 531
- subscription-set members
  - adding 75, 195, 376
  - applying subset of columns 91
  - applying subset of rows 91
  - defining target key 93
  - mapping between columns 92
  - mapping data types 92
  - multi-tier replication 85
  - number per subscription set 64
  - removing 423
  - selecting target types 78
  - update-anywhere replication 87
- subsetting
  - advanced techniques
    - during registration 107
    - using predicates 109
    - using triggers on CD tables 108
    - using views 108
  - columns at target 91
  - registered columns 42
  - registered rows of changes 43
  - rows of changes at target 91
- suspending
  - Capture program
    - for UNIX 136, 322
    - for Windows 136, 322
    - for z/OS 136, 322
- Sybase
  - replication restrictions 45
- SYSADM 18, 19
- system change journal management 34
- system-started tasks 454

**T**

- table structures 471
- table-mode processing 7, 72
- tables
  - adding columns 185
  - ALERTS (Monitor alerts) 533
  - APPENQ (Apply enqueue) 513
  - APPLY\_JOB (Apply job) 513
  - APPLYTRACE (Apply trace) 514
  - APPLYTRAIL (Apply trail) 515
  - at Apply control server 513
  - at Capture control server 483
  - at Monitor control server 533
  - at target server 541



tables (*continued*)

- AUTHTKN (Apply-qualifier cross-reference) 484
- base aggregate 541
- CAPENQ (Capture enqueue) 485
- CAPMON (Capture monitor) 236, 486
- CAPPARMS (Capture parameters) 487
- CAPSCHEMAS (Capture schemas) 483
- CAPTRACE (Capture trace) 236, 490
- CCD (consistent-change-data)
  - Capture control server 491
  - target server 543
- CD (change-data) 493
- change aggregate 542
- changing attributes 184
- CONDITIONS (Monitor conditions) 534
- conflict detection for 11
- CONTACTGRP (Monitor group contacts) 537
- CONTACTS (Monitor contacts) 537
- control tables
  - connectivity failure recovery 238
  - creating 23
  - dynamic 231
  - I/O error recovery 238
  - maintaining 231
  - pruning 233
  - reorganizing 232
  - RUNSTATS utility 231
  - static 233
- deactivating 188
- GROUPS (Monitor groups) 538
- maintaining CCD tables 61
- MONENQ (Monitor enqueue) 538
- MONSERVERS (Monitor servers) 539
- MONTRACE (Monitor trace) 540
- point-in-time 546
- PRUNCNTL (pruning control) 494
- PRUNE\_LOCK (prune lock) 496
- PRUNE\_SET (prune set) 496
- reactivating 189
- REG\_EXT (register extension) 497

tables (*continued*)

- REG\_SYNCH (register synchronization) 505
- REGISTER (register) 498
- registering
  - DB2 37
  - non-DB2 relational 39
  - procedure 183
- removing registrations 190
- replica 11, 546
- RESTART (restart) 505
- SEQTABLE (sequencing) 507
- SIGNAL (signal) 507
- stop capturing changes 188
- structures 471
- SUBS\_COLS (subscription columns) 519
- SUBS\_EVENT (subscription events) 521
- SUBS\_MEMBR (subscription members) 158, 522
- SUBS\_SET (subscription sets) 526
- SUBS\_STMTS (subscription statements) 531
- target tables
  - See also* target tables
  - maintaining 239
- UOW (unit-of-work) 510
- user copy 547
- target indexes 93
- target keys 93
- target servers
  - log impact 7
  - tables at 541
- target tables
  - applying subset of columns 91
  - applying subset of rows 91
  - base aggregate
    - definition 78
    - structure 541
    - usage 81
  - CCD (consistent-change-data)
    - overview 78
    - structure 543
  - change aggregate
    - definition 78
    - structure 542
    - usage 82
  - defining columns 91
  - defining rows 91
  - defining target key 93
  - fragmenting 91
  - list of 541
  - maintaining 239

target tables (*continued*)

- mapping to sources 75
- new columns for 113
- point-in-time
  - definition 78
  - structure 546
  - usage 81
- replica
  - conflict detection for 11
  - definition 78
  - structure 546
  - usage 87
- storage requirements 8
- table structures, quick reference 482
- user copy
  - definition 78
  - structure 547
  - usage 80
  - user defined 80, 89
- target-key columns
  - updating 94
- targets
  - forcing full refresh 263
  - profiles 250
- term parameter (for Apply) 148, 314
- term parameter (for Capture) 131, 321, 327
- termination characters, in generated SQL scripts 116
- three-tier replication
  - configuration 85
- throughput
  - Apply program 167
  - Capture program 165
- throughput rates
  - Capture triggers 12
- time-based scheduling 73
- tips
  - checking if Apply processed a set successfully 148
  - deleting rows from the Apply trail table 149
  - estimating use of space 6
  - using sleep versus copyonce parameters 143, 146
  - using stored procedures for additional processing of sets 152
  - using stored procedures with ASNDONE 153
  - verifying a service is set up correctly 459

- tips (*continued*)
  - verifying change capture began 122
- trace facility
  - for OS/400 446
  - for UNIX 341
  - for Windows 341
  - for z/OS 341
- TRACE parameter 431
- trace\_limit parameter
  - overview 131
  - pruning from the Replication Alert Monitor 175
  - use with asncap command 321
  - use with asncmd command 328
  - use with asnmon command 333
- transaction throughput rates
  - Capture triggers 12
- transaction-mode processing 7, 72
- transactions
  - memory used by 3
- transforming data
  - at registration 111
  - at subscription 112
  - creating computed columns 113
  - renaming columns 113
- translating data 14
- TRCLMT parameter 441
- triggers
  - capturing data 12
  - merging 13
  - on CD tables 108
  - suppressing data capture 108
- trlreuse parameter 148, 313
- TRLREUSE parameter 434
- troubleshooting commands
  - asntrc 341
  - WRKDPTRC 446
- TSO 453
- tuning
  - commit\_interval parameter 4
  - memory\_limit parameter 4
- U**
  - Unicode tables 641
  - unit-of-work (UOW) table
    - columns in CCD tables 83
    - pruning 235, 510
    - storage requirements 9
    - structure 510
  - UOW (unit-of-work) table
    - columns in CCD tables 83
    - pruning 235, 510
    - storage requirements 9
  - UOW (unit-of-work) table (*continued*)
    - structure 510
  - UOW\_CD\_PREDICATES
    - column 109
  - update database configuration
    - command 27
  - update-anywhere replication
    - conflict detection
      - overview 54
      - planning for 11
      - requirements 46, 54
    - defining subscription sets 87
    - fragmentation for 11
    - recapturing changes 49
  - updated primary key columns 48
  - updates
    - as deletes and inserts 48
    - conflicts 54
  - user copy table
    - definition 78
    - structure 547
    - usage 80
  - user IDs
    - authorization 19
    - for Apply program 21
    - for Capture program 19
    - for Capture triggers 20
    - for Replication Alert Monitor 22
    - password files 23
  - user IDs for Replication Center 247
  - USER parameter 429
  - USER signals 214
  - user-defined data types 97
  - user-defined tables 80, 89
- V**
  - VALIDPROC clauses 97
  - vertical (column) subsetting
    - at the source 42
    - at the target 91
  - views
    - changing attributes 184
    - registering
      - as sources 60
      - overview 58
      - procedure 183
      - restrictions 58, 60
      - using correlation ID 58
- W**
  - WAIT parameter 439
  - warm start, Capture program
    - for OS/400 438, 445
    - for UNIX 130, 320
  - warm start, Capture program (*continued*)
    - for Windows 130, 320
    - for z/OS 130, 320
  - warmns startmode 130
  - warmsa startmode 130
  - warmsi startmode 130
  - WHERE clause
    - PREDICATES column
      - restriction 109
      - row subsets 91
  - Windows Service Control Manager (SCM) 459
  - Windows services names 302
  - work management objects 34
  - WRKDPTRC command 446
  - WRKJOB command 163
  - WRKREGINF command 36
  - WRKSBJOB command 163
  - WRKSBSJOB command 163
- Z**
  - z/OS server
    - connecting to 16

---

## Notices

IBM may not offer the products, services, or features discussed in this document in all countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country/region or send inquiries, in writing, to:

IBM World Trade Asia Corporation  
Licensing  
2-31 Roppongi 3-chome, Minato-ku  
Tokyo 106, Japan

**The following paragraph does not apply to the United Kingdom or any other country/region where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make

improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product, and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information that has been exchanged, should contact:

IBM Canada Limited  
Office of the Lab Director  
8200 Warden Avenue  
Markham, Ontario  
L6G 1C7  
CANADA

Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems, and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements, or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious, and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

#### COPYRIGHT LICENSE:

This information may contain sample application programs, in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (*your company name*) (*year*). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. *\_enter the year or years\_*. All rights reserved.

---

## Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both, and have been used in at least one of the documents in the DB2 UDB documentation library.

ACF/VTAM	LAN Distance
AISPO	MVS
AIX	MVS/ESA
AIXwindows	MVS/XA
AnyNet	Net.Data
APPN	NetView
AS/400	OS/390
BookManager	OS/400
C Set++	PowerPC
C/370	pSeries
CICS	QBIC
Database 2	QMF
DataHub	RACF
DataJoiner	RISC System/6000
DataPropagator	RS/6000
DataRefresher	S/370
DB2	SP
DB2 Connect	SQL/400
DB2 Extenders	SQL/DS
DB2 OLAP Server	System/370
DB2 Universal Database	System/390
Distributed Relational Database Architecture	SystemView
DRDA	Tivoli
eServer	VisualAge
Extended Services	VM/ESA
FFST	VSE/ESA
First Failure Support Technology	VTAM
IBM	WebExplorer
IMS	WebSphere
IMS/ESA	WIN-OS/2
iSeries	z/OS
	zSeries

The following terms are trademarks or registered trademarks of other companies and have been used in at least one of the documents in the DB2 UDB documentation library:

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.





---

## Contacting IBM

In the United States, call one of the following numbers to contact IBM:

- 1-800-237-5511 for customer service
- 1-888-426-4343 to learn about available service options
- 1-800-IBM-4YOU (426-4968) for DB2 marketing and sales

In Canada, call one of the following numbers to contact IBM:

- 1-800-IBM-SERV (1-800-426-7378) for customer service
- 1-800-465-9600 to learn about available service options
- 1-800-IBM-4YOU (1-800-426-4968) for DB2 marketing and sales

To locate an IBM office in your country or region, check IBM's Directory of Worldwide Contacts on the web at [www.ibm.com/planetwide](http://www.ibm.com/planetwide)

---

## Product information

Information regarding DB2 Universal Database products is available by telephone or by the World Wide Web at [www.ibm.com/software/data/db2/udb](http://www.ibm.com/software/data/db2/udb)

This site contains the latest information on the technical library, ordering books, client downloads, newsgroups, FixPaks, news, and links to web resources.

If you live in the U.S.A., then you can call one of the following numbers:

- 1-800-IBM-CALL (1-800-426-2255) to order products or to obtain general information.
- 1-800-879-2755 to order publications.

For information on how to contact IBM outside of the United States, go to the IBM Worldwide page at [www.ibm.com/planetwide](http://www.ibm.com/planetwide)



Part Number: CT191NA



Printed in the United States of America  
on recycled paper containing 10%  
recovered post-consumer fiber.

SC27-1121-00



(1P) P/N: CT191NA



Spine information:



IBM<sup>®</sup> DB2 Universal Database<sup>™</sup> DB2 Replication Guide and Reference

Version 8